R-573-NASA/PR
May 1971

COMPUTER PERFORMANCE ANALYSIS: APPLICATIONS OF ACCOUNTING DATA

R. Watson

A Report prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
AND
UNITED STATES AIR FORCE PROJECT RAND

Rand
SANTA MONICA, CA. 90406

This research is sponsored by the National Aeronautics and Space Administration under Contract No. NAS-12-21-44 and the United States Air Force under Project Rand — Contract No. F44620-67-C-0045 — monitored by the Directorate of Operational Requirements and Development Plans, Deputy Chief of Staff, Research and Development, Hq USAF. Views or conclusions contained in this study should not be interpreted as representing the official opinion or policy of Rand, NASA or of the United States Air Force.

R-573-NASA/PR

May 1971

COMPUTER PERFORMANCE ANALYSIS: APPLICATIONS OF ACCOUNTING DATA

R. Watson

A Report prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

AND

UNITED STATES AIR FORCE PROJECT RAND

Rand

SANTA MONICA, CA. 90406

Rand maintains a number of special subject bibliographies containing abstracts of Rand publications in fields of wide current interest. The following bibliographies are available upon request:

Africa • Arms Control • Civil Defense • Combinatorics
Communication Satellites • Communication Systems • Communist China
Computing Technology • Decisionmaking • Delphi • East-West Trade
Education • Foreign Aid • Foreign Policy Issues • Game Theory
Health-related Research • Latin America • Linguistics • Maintenance
Mathematical Modeling of Physiological Processes • Middle East
Policy Sciences • Pollution • Program Budgeting
SIMSCRIPT and Its Applications • Southeast Asia • Systems Analysis
Television • Transportation • Urban Problems • USSR/East Europe
Water Resources • Weapon Systems Acquisition
Weather Forecasting and Control

To obtain copies of these bibliographies, and to receive information on how to obtain copies of individual publications, write to: Communications Department, Rand, 1700 Main Street, Santa Monica, California 90406.

Published by The Rand Corporation

PREFACE

This report suggests that analysis of computer system accounting data can be a valuable tool in computer performance analysis. The study describes the data available from computer accounting systems and the techniques for conditioning and reducing the data. It then discusses various applications for using the data in the measurement and evaluation of computer system performance.

Virtually all Air Force and NASA computer installations collect and record accounting data (which, in computer systems, are an account of computer resources used by each job processed). However, seldom are these data used except at those installations that utilize them to charge for computer services.

This report emphasizes the applications of accounting-data analysis that can be performed at low marginal cost and suggests that Air Force and NASA computer installations take full advantage of such analyses in economizing and/or improving the performance of their computer systems.

Although this study was initially funded by NASA, its applicability to Air Force computer systems led to continuation of the work under Project Rand.

SUMMARY

Virtually all Air Force and NASA computer installations collect and record accounting data (which, in computer systems, are an account of computer resources used by each job processed). However, seldom is any use made of these data except at those installations that use accounting data to charge for computer services. This report suggests that the analysis of computer system accounting data can be a valuable tool in computer performance analysis.

The report describes the types of accounting data generally available at most computer installations--discussed in the context of the accounting data collected by Rand's IBM 360/65 computer system. Techniques for conditioning and reducing the data are then discussed, along with various reports that can be generated from such data. The balance of the report concerns specific applications of accounting-data analysis in computer performance analysis.

The most heavily stressed application is the use of accounting data to measure the effect of a system modification. The volume of accounting data available and the fact that they are generated from "typical" workloads make it possible for carefully structured multiple regression and cluster analyses to produce definitive conclusions on computer system performance. This application is exemplified by analyzing the effects (changes in system throughput and efficiency) of a core augmentation to Rand's computer system.

The report also discusses such other applications of accounting-data analysis as validation of system-performance measurements taken by hardware or software monitors, and use in either developing and testing a new computer charging scheme or updating an installation's current charging scheme in the face of changing workloads or changes in the system.

ACKNOWLEDGMENTS

The author wishes to thank Dr. Carl Morris of the Rand Mathematics Department for his help and guidance in performing the statistical analyses discussed in this report.

CONTENTS

FIGURES

TABLES

I. INTRODUCTION

BACKGROUND

Analysis of computer system performance can lead to big savings for a computer installation. Bottlenecks in throughput can be located and significantly reduced by balancing the load over many system components instead of one or two. The purchase of a more powerful (and more costly) computer system may be temporarily or indefinitely postponed by tuning the present system so that it runs more efficiently, thereby increasing its throughput. Turnaround time can be reduced to decrease the time users spend waiting for results. These are only a few of the benefits that can be derived from computer system performance analysis.

Such analysis can often be done most cost-effectively with an existing source of machine-readable data: accounting data. At present, however, very little use is made of computer system accounting data to measure and evaluate computer system performance. Three reasons seem to account for this:

1. Computer system measurement and evaluation is still a very new field; many computer installations simply do not measure the performance of their systems.
2. Computer installations that do measure and evaluate computer performance have not been exposed to the potential value of accounting data and use the more publicized tools--analytic or simulation models or hardware and software monitors.
3. The analysts doing the measuring either do not know how to use accounting data, doubt their applicability, or fear that they will have to spend a great deal of time conditioning and reducing the data before they can even begin to analyze them.

This report suggests that accounting-data analysis can be a valuable tool in measuring and evaluating computer system performance. The study describes the data available for accounting systems and the methods of conditioning and reducing these data. It then discusses various applications for using the data in the measurement and evaluation of computer system performance.

ACCOUNTING DATA

Computer system accounting data are an account of the computer system resources used in processing jobs. They are primarily used in charging users for computer resources, but they can also be useful in measuring computer system performance and workloads processed. The data are usually broken down into computer resources used per job (or per job step within a job).

Accounting data generally consist of three types of information:

1. *Identification data*, which may include programmer identification, a project number, or any other identification information desired by a particular installation;
2. *Quantities of computer system resources used*, such as Central Processor Unit (CPU) seconds used, the number of Input/Output (I/O) accesses made to peripheral storage equipment, and the amount of core storage requested;
3. *The initiation and termination times* of the interval over which later resources were used.

In addition to the accounting data created and recorded by the computer system, many installations maintain a sign-in log to gain a measure of turnaround time. The accounting system and sign-in log in use at The Rand Corporation are detailed in Sec. II.

RECORDING ACCOUNTING DATA

Computer system resources used by a job are accumulated from the time job processing starts until it terminates. The precise time at which the accounting system designates a job has started or terminated varies from computer system to computer system and accounting system to accounting system. For example, "start time" recorded by the accounting system for Rand's IBM 360/65 computer system corresponds to the time the requested amount of main memory is allocated to a job. "Termination time" corresponds to the time a job has both completed processing and been unloaded from main memory. The accumulated accounting data are printed on the job's output and recorded in temporary storage

(usually disk) until the end of the day, when they are permanently recorded on either magnetic tape or punched cards.

WIDELY USED MEASUREMENT TOOLS

Computer system measurement and evaluation has only recently emerged as a separate field in computer science. This emergence has been spurred by the development and refinement of sophisticated tools for measuring computer system activity and performance. Hardware and software monitors are the most widely used tools; they are briefly described below.†

Hardware Monitors

A hardware monitor has a set of high-impedance probes that attach directly to the computer's circuitry. Each probe generates a signal upon activation of the circuits to which it is attached; the signals are then fed to counters or timers. The most attractive feature of hardware monitors is their ability to obtain accurate, high-resolution data without affecting the performance of the host system. Their disadvantage is that they are limited to checking only a few items (channel usage, disk movements, etc.) at one time. Although some monitors have in excess of 100 probes, usually ". . . a relatively small number of counters/timers are available (often 16), and only a small number of probes can be used (often 20)."‡ Another disadvantage is the difficulty of tracking the computer resources used by each job processed by the system.

Software Monitors

A software monitor consists of code residing in the core of a computer. It functions as a monitor that is allowed to break into the system and collect data on hardware usage, disk-head movement, supervisor routines used, etc. A high-density sampling method is generally

---

†See Ref. 1 for an in-depth discussion of computer system measuring devices.

‡Ref. 1, p. 16.

used in which the system is interrupted at periodic intervals (usually 1/60 sec or a multiple thereof). Software monitors are often preferred to hardware monitors because many more types of data can be collected at one time. Their primary disadvantage is that they effect an appreciable load on the system--as great as 20 percent on CPU activity and 10 percent on I/O activity.[†]

USE OF ACCOUNTING-DATA ANALYSIS AS A MEASUREMENT TOOL

Accounting data record the quantities of system resources used by each job rather than the utilization of each system element. As such, they are more suited for evaluating workload characteristics than for evaluating computer system performance. Nonetheless, accounting data can be used to measure computer system performance by analyzing such gross performance measures as the degree to which main memory is used (core utilization), the percent of total CPU cycles available that are used for processing jobs (CPU utilization), the average number of I/O accesses processed per second, and the average revenue produced per hour. Section III coordinates these general uses into specific uses for measuring and evaluating computer system performance.

[†]Ref. 1, p. 19.

II. ACCOUNTING DATA: DESCRIPTION AND RELATED DATA-REDUCTION ACTIVITIES

INTRODUCTION

As an example of the types of data that can be collected by computer accounting systems and the data-reduction and data-analysis efforts necessary to convert these data into meaningful information, this section discusses the accounting-data analysis performed at Rand. The accounting data collected from Rand's IBM 360/65 computer system are described and the data-reduction activities performed on the data are discussed.

In order for the general reader to follow the discussion, he must understand the operation of this particular computer system. Therefore, some background information is provided that (1) describes the multiprogramming environment in which the computer system operates, (2) defines the various terms used in measuring system usage, and (3) discusses the differentiation between jobs and job steps made by the system, and describes how this differentiation affects the accounting system.

Multiprogramming Environment

Multiprogramming is most simply defined as the concurrent processing of two or more jobs. Although only one job is using the CPU at any one time, other jobs can be allocated memory segments and be using peripheral devices (printers, card reader, tapes, disks, etc.) simultaneously. The operating system gives control (use of the CPU) to one job and then switches it to another job whenever either (1) the one in control becomes idle, i.e., whenever it is waiting for a peripheral device to complete some I/O processing, or (2) a higher-priority job enters the system. Thus, multiprogramming capitalizes on the fact that most jobs do not use all the memory or all I/O devices. The operating system loads several jobs into memory in an attempt to maximize the efficiency of the computer system by keeping the CPU, memory, and I/O devices busy at all times.

To offset the relative slowness of the basic I/O devices (card reader, card punch, and printers), a job never directly reads or punches cards, or prints output. Instead, the system reads input cards onto a disk-storage file at the time the job is submitted and stores the information on disk until the job is scheduled to begin execution. During execution, the job reads the input information directly from the faster disk-storage devices. Similarly, any output generated by the job is written on a disk file. After the job has terminated, the system prints or punches (onto cards) all output that the job stored on the disk file. This buffering of input and output onto a disk-storage device is called "spooling." The tasks that perform both spooling and all other overhead functions are called "system tasks." These are accounted for separately from user jobs.

Terms Used in Measuring System Usage

Generally, the basic terms used to describe the system resources used by a job are (1) CPU seconds used, (2) elapsed time (clock time) on the system, (3) memory requested (in K-bytes, where K equals 1024), and (4) I/O usage. The definitions of the first three terms are self-evident. But I/O usage, because its measurement varies so widely from installation to installation, requires further explanation. In accounting for I/O usage, many computer installations include number of cards read, number of cards punched, number of lines printed, and other I/O operations. Many other installations, including Rand, use only the number of EXCPs (EXecute Channel Program), that is, the number of requests made by a job (or system task) to I/O devices. Because an EXCP is equivalent to an I/O request (or I/O access), EXCPs are henceforth referred to as I/Os.

Jobs and Job Steps

Each job is made up of one or more job steps. A typical example is a three-step job consisting of (1) a "compile" step, which translates a source program into an object module of machine instructions; (2) a "Linkedit" step, which resolves external references and gathers

additional subroutines from subroutine libraries and public libraries; and (3) a "GO" step, during which computation is performed. After the operating system introduces a job to the computer system, it no longer deals with the job as a whole, but with the job steps. Thus, the accounting data record the resources and services used by each job step and accumulate the resources used by each to determine the total resources used for each job.

AVAILABLE ACCOUNTING DATA

All data generated by a computer system for accounting are also available for measuring and evaluating the computer system. In actual fact, only a portion of the data collected is used for charging purposes, although practically all the data are useful in measuring and evaluating the computer system. Generally, the accounting data are of two different forms--system accounting data and sign-in log data. System accounting data usually consist of records created at the end of each job step and/or at the end of each job. These data primarily consist of job-identification data plus the quantities of system resources used. The sign-in log usually contains brief identification information and sign-in and sign-out times for each job submitted to the system.

System Accounting Data

The types of system accounting data collected by computer systems vary considerably from installation to installation. The types of data collected depend upon (1) the make of the computer system, (2) the operating system under which the system is running (and also the version that is currently being used), and (3) any additional data-collection routines implemented into the operating system by installation systems programmers.

At the time the research for this study was performed, Rand's IBM 360/65 was operating under Release 17 of OS/360. The only standard accounting data generated by this release of OS/360, however, were job

identification and the number of CPU seconds used by each job.† Therefore, Rand systems programmers inserted numerous modifications in the operating system to increase the accounting data collection capabilities. The modified accounting system generates two separate (but similar) logs of accounting data: an "accounting log" and a "system log."

Accounting Log. In generating the accounting log, the system generates a record for each job step processed by the computer system during a day's operations. The records are temporarily stored on disk, and are punched onto cards (one record per card) at the end of the day. Each record in the accounting log contains the following information:

- Date;
- Job number (a unique number assigned to each research project for charging purposes);
- Man number (identification of programmer submitting the job);
- Job-step number (within each job, the job steps are sequentially numbered by the system);
- Core (memory) requested;
- CPU seconds used;
- Job I/Os used;
- Time on;
- Time off.

System Log. The system log generates a record, similar to accounting-log records, for each job step. However, the system log also generates a record each time a system task either starts or ends. Whereas the accounting log at Rand is used primarily as a basis for charging computer users, the system log is used to generate monthly reports of system usage. However, the system log has also been used as a backup for faulty or lost accounting-log records. Each record in the system log contains the following information:

For job steps:

- Date;
- Man number;

---

†Later releases of OS/360 (i.e., Release 18 and Release 19) can be equipped with a much more extensive accounting system. This new accounting system, written by IBM, is called System Management Facilities (SMF) [2]. It collects accounting data similar to those collected by Rand's modified system, plus additional data. The additional data collected by SMF are described later in this section.

- Job-step number;
- Core (memory) requested;
- Core (memory) used;
- CPU seconds used;
- Job I/Os used (number of I/O accesses);
- Processor program used (FORTRAN compile, Linkedit, etc.);
- Time on;
- Time off.

For system tasks:

- Date;
- Identification of system task;
- Time that task started or ended;
- Accumulated system I/Os (for all system tasks) that were performed since the previous system task record was generated. Excludes job-step I/Os.

Sign-in Log

Most computer installations require users to fill out a sign-in log each time they submit a job. Rand not only maintains a sign-in log, but also punches the sign-in log data onto cards for future use in generating reports. The following data are logged for each job that enters Rand's IBM 360/65 computer system; similar data are usually collected at other computer installations:

- Date;
- Sequence number;
- Sign-in time;
- Job number;
- Man number and program identification;†
- Job class (scheduling priority);
- Sign-out time (filled in by operators after job leaves the system).

In contrast to the accounting-log data and the system-log data, Rand's sign-in log data are not permanently stored. Instead, they are stored one month at a time and then summarized into a monthly report of computer operations. Therefore, original sign-in log data at Rand are not available for analyzing computer operations further back than one month.

---

†Punched onto cards only during selected periods.

Expanded Accounting Systems

The accounting data described above (from the accounting log, the system log, and the sign-in log) are representative of the data that can be collected at most computer installations. All these data items are either automatically collected by vendor-supplied accounting systems or can be collected by making minor modifications to the accounting system. A number of installations have modified their operating systems to collect much more data than are described above. Furthermore, computer manufacturers--recognizing the value of accounting data--are now supplying more sophisticated accounting systems with their operating systems.

An expanded accounting system recently made available is IBM's SMF accounting package [2] (available in versions 18 or higher of OS/360). This system, which is more sophisticated than those currently in use, records all data recorded by Rand's current accounting system, plus additional data. Some of the additional data recorded by SMF are listed below:

Additional data recorded per job step:

- Completion or condition codes for abnormal job terminations.
- Number of data sets used.
- Number of tapes used.
- Number of disks used.
- Number of input cards read.
- Attach time (time that initiator picked up job step from input queue).
- Number of I/Os per data set.

Additional data recorded from summary of day's operations:

- Total CPU "wait" time.
- Number of IPLs[†] and time of each IPL.

[†]An Initial Program Load (IPL) involves loading the operating system into the computer system. An IPL is performed when the system is started at the beginning of the day and also each time it is started after abnormally going down.

- o Number of jobs processed.
- o Number of job steps processed.
- o Number of data sets and records written from the printer.

REDUCTION OF ACCOUNTING DATA

Reducing computer accounting data involves conditioning the raw data into processible form and then processing them to obtain meaningful measurements of system performance. The techniques used to condition and process the accounting data from Rand's IBM 360/65 computer system are described below.

Data Conditioning: Error Correction

One of the most important steps in reducing the accounting data is the detection of any bad data and subsequent correction or deletion of, or allowance for all such data. If this is not done, the data may be misinterpreted.

If the accounting data for a computer system are used for billing purposes, most obvious errors or discrepancies have very likely been corrected. Such is the case with the accounting-log data at Rand. Although the data are reviewed primarily to screen out invalid project numbers, this reviewing process also detects and eliminates bad or void records.

Accounting data not used for billing purposes, such as the system log at Rand, are seldom edited for incorrect data. Therefore, eventual use of such data requires conditioning in the form of checking for errors and correcting (or allowing for) any errors detected. Most errors at Rand occur as duplicate or bad (garbage) records and are caused by procedural problems. For instance, a bug in the procedure that generated the system log caused the generation of numerous duplicate records and the deletion of blocks of data. The system was generating so many duplicates of each record that an entire reel of magnetic tape was filled up after recording data for only three to five days.† Customer engineers

†Three to five days of system-log data should only have used about 75 ft of tape.

from the computer manufacturer and Rand systems programmers tried to solve the problem. However, all efforts to correct the situation failed and, in order to use the data, special code in the conditioning program had to be written to eliminate the duplicate records. However, nothing could be done about the blocks of data deleted. Rand systems programmers finally solved this problem. However, errors occasionally occur in all accounting systems due to the complexity of modern operating systems; provisions must be made in the conditioning process to detect and correct (or allow for) such errors.

Another error that occasionally occurs in the generation of accounting data is assigning the wrong date to a portion of a day's data. This occurs when the computer operators forget to reset the date at the beginning of the day (or set it incorrectly). Although such errors are soon corrected, all accounting data generated up to the time of correction retain the wrong date. Such errors must be detected and corrected in conditioning programs.

Data Conditioning: Pre-processing

After the obvious inaccuracies in the accounting data are removed, the data must be conditioned for efficient input to analysis programs. For the accounting log at Rand, this simply entailed converting all past data from cards to magnetic tape. Three months of accounting data were stored per tape and the data on each tape were further divided into files of one week each. This method of storage was chosen over an alternate method--storing an entire year's data on one or two tapes (with no subdivision into files)--because it allowed a particular day or week of data to be accessed quickly and easily by indicating the proper tape and file number.

A similar readying process was applied to the system-log data, and they were stored on tape in the same format as the accounting log. The readying process for the system log differed in that duplicate and invalid records were detected and eliminated while the data were reformatted.

Processing the Accounting Data

Once the accounting data were conditioned and stored in an easily accessible form, they could be processed into meaningful information. As discussed earlier, the system log recorded the same data as the accounting log (except for "Job Number") *plus* the system processing programs used by job steps and the I/Os used by system tasks. However, system-log data had not been recorded during much of the period over which measurement and evaluation of the computer system was desired.† Thus, most of the analysis was performed on the accounting-log data.

Analysis programs were written to process the accounting data. The programs were written with two objectives in mind:

1. Extract or generate as many different types of information as possible from the data.
2. Print out the information on an easily readable report that could be useful to the director of a computer center.

Analysis of System-Log Data. Although very little system-log data were available during the period of computer operations analyzed in this report, a program to analyze the data was written for use in future analyses. This program's primary purpose is to collect and record the number of system I/Os processed by the system. These data can then be combined with the job I/Os (calculated by a program that analyzes the accounting log) to determine the total I/Os processed by the system. This same program computes workload characteristics in terms of which system-processing programs were used over a given time interval and which resources were used by the job steps processed by the programs. A report is generated that includes (1) a frequency distribution of the number of job steps that used each processing

†The problem of recording excessive duplicate records on the system-log tapes resulted in the usage of so many tapes that the recording of system-log data was discontinued until the problem could be solved. As such, very little system-log data were available for the first year or so of operations (of the 360/65). Unfortunately, this was the same period over which most of the accounting-data analysis reported in this study was done.

program, and (2) information pertaining to the number of CPU seconds and I/Os used. Figure 1 illustrates an example of this report.

Analysis of Accounting-Log Data. Two programs were written to analyze the accounting-log data. The primary purpose of the first program was to extract and record any time intervals during which the CPU was either "idle" or "down." Such information was necessary because performance measures (e.g., CPU utilization, I/Os processed per hour) were calculated by summing the total resources used over a specified time interval and dividing by the "net" time the CPU was processing. The idle or down times corresponded to times during which main memory (core) was vacant of jobs. These times were determined from the "on time" and "off time" for the job steps. A table was printed (Fig. 2) containing (1) the clock time of the last job that left core before a vacant period (BEGIN OF INTERVAL), (2) the clock time of the first job that entered core to end the vacant interval (END OF INTERVAL), and (3) the length of the vacant interval (INTERVAL LENGTH).

In addition to recording idle and down times, the first program maintained the number of jobs in memory and the amount of memory (core) in use during the time span that the program analyzed the accounting data. It also accumulated the number of CPU seconds used, the number of job I/O requests, and the number of job steps processed throughout the time span analyzed. This information could be printed and/or plotted, depending upon options specified in running the program. The Appendix contains graphs, generated by this program, of system resources used.

A second program uses the idle time and down time information collected from the first program and further analyzes the accounting log to calculate measures of performance for a given time period. It also determines various workload characteristics for the same time period.

This program generates a report of identification information and system resources used by each job step. A portion of this report for October 28, 1969 is shown in Fig. 3. The portion of the report illustrated includes all job steps with a "time on" between 8:32:23 a.m. and 9:24:00 a.m., whereas the original report includes all job steps during the 9-hr time period between 8:30 a.m. and 5:30 p.m. (17:30 on a 24-hr clock).

REPORT OF PROCESSING PROGRAM USAGE BY JOB STEP

DATE = FEB. 24, 1970 TIME INTERVAL = 0900 - 1200

PROCESSING PROGRAM USED	TOTAL JOB STEPS	AVERAGE CORE REQUESTED	TOTAL CPU SECS USED	TOTAL I/O'S USED	# OF JOB STEPS THAT RAN	# OF JOB STEPS THAT DION'T RUN	AVE CPU SECS USED (ALL JOB STEPS)	AVE CPU SECS USED (ONLY FOR STEPS THAT RAN)	AVE I/O'S USED (ALL JOB STEPS)
NO NAME	10	91K	0.0	539.	0	10	0.0	0.0	54.
BMD05D	2	156K	65.4	5393.	2	0	32.7	32.7	2697.
B8750TND	1	52K	4.3	671.	1	0	4.3	4.3	671.
CAPMOD2	3	228K	19.2	3065.	3	0	6.4	6.4	1022.
CHK4060	5	52K	1.2	415.	4	1	0.2	0.3	83.
CLOSE	5	52K	1.2	419.	4	1	0.2	0.3	84.
DAGO1	8	174K	607.3	74692.	8	0	75.9	75.9	9337.
DISTLBL	1	104K	3.2	1246.	1	0	3.2	3.2	1246.
D0155S81	1	200K	234.0	4481.	1	0	234.0	234.0	4481.
F7680TDP	1	52K	0.8	126.	1	0	0.8	0.8	126.
GO	55	99K	934.4	99871.	55	0	17.0	17.0	1816.
HARRIET	1	52K	3.9	2024.	1	0	3.9	3.9	2024.
H1200UTL	4	54K	2.2	882.	3	1	0.5	0.7	221.
H3250JH	3	150K	291.4	3750.	3	0	97.1	97.1	1250.
H3790CEJ	1	90K	30.5	2287.	1	0	30.5	30.5	2287.
IEBGENER	2	52K	1.9	489.	2	0	0.9	0.9	245.
IEBPTPCH	4	52K	24.1	8324.	4	0	6.0	6.0	2081.
IEBUPDTE	2	52K	1.2	571.	2	0	0.6	0.6	286.
IEFBR14	12	52K	133.6	595.	6	6	11.1	22.3	50.
IEHMOVE	11	75K	42.1	15114.	9	2	3.8	4.7	1374.
IEHPROGM	5	84K	2.0	364.	3	2	0.4	0.7	73.
IEJFAAA0	1	64K	2.4	467.	1	0	2.4	2.4	467.
IEKAA00	1	228K	28.7	709.	1	0	28.7	28.7	709.
IEMAA	16	100K	130.2	8489.	15	1	8.1	8.7	531.
IEQCBL00	14	104K	95.7	9990.	14	0	6.8	6.8	714.
IERRC000	16	62K	87.8	29106.	14	2	5.5	6.3	1819.
IEWL	30	102K	54.8	13274.	26	4	1.8	2.1	442.
IEYFORT	37	114K	541.2	27444.	36	1	14.6	15.0	742.
IHGUAP	4	79K	7.7	3622.	2	2	1.9	3.9	906.
LINKEDIT	45	100K	67.7	21120.	38	7	1.5	1.8	469.
MAIN	1	52K	0.4	101.	1	0	0.4	0.4	101.
MARKIV	8	79K	17.3	3860.	4	4	2.2	4.3	483.
MASM	2	104K	3.6	749.	2	0	1.8	1.8	375.
MULTASM	7	78K	142.9	25225.	5	2	20.4	28.6	3604.
M9418NMS	2	136K	20.6	736.	2	0	10.3	10.3	368.
RJSTATUS	7	52K	0.0	353.	0	7	0.0	0.0	50.
SCRTCHEM	5	52K	1.7	813.	5	0	0.3	0.3	163.
SIM2	3	188K	197.4	1662.	3	0	65.8	65.8	554.
SPSS	3	219K	49.6	5429.	3	0	16.5	16.5	1810.
SUPERZAP	2	78K	1.3	518.	2	0	0.6	0.6	259.
TCRGENER	2	104K	2.8	431.	2	0	1.4	1.4	216.
TSRDUP	2	53K	5.2	234.	1	1	2.6	5.2	117.
TSREDIT	2	52K	15.2	6498.	2	0	7.6	7.6	3249.
TSRLIST	1	52K	20.7	11105.	1	0	20.7	20.7	11105.
W7888STS	2	166K	12.5	1181.	2	0	6.2	6.2	591.
	350		3911.3	398434.	296				

Fig. 1--Report of Processing Program Usage by Job Step

TIME INTERVALS DURING WHICH THE CPU WAS EITHER DOWN OR IDLE

DATE = 102369

BEGIN OF INTERVAL	END OF INTERVAL	INTERVAL LENGTH
8.29	8.38	0.09
8.40	8.42	0.03
11.88	12.14	0.26
12.18	12.22	0.04
12.26	12.47	0.21
15.43	15.48	0.05
17.79	17.97	0.17
18.40	18.44	0.04
18.44	18.49	0.05
18.50	18.57	0.07
18.58	18.64	0.06
18.64	18.73	0.08
18.73	18.76	0.03
18.76	18.86	0.10
18.88	18.93	0.05
18.94	18.95	0.02
18.96	19.96	1.00
20.01	20.16	0.15
22.29	22.30	0.01
22.34	22.35	0.01
22.47	22.49	0.01
22.53	23.55	1.01
23.91	23.95	0.04
	TOTAL =	3.60

Fig. 2--CPU Idle and Down Times

JOB STEP ACCOUNTING REPORT FOR OCT 28, 1969

TIME ON	TIME OFF	PROGRAMMER I.D.	STEP # WITHIN JOB	CORE REQ'D	CPU SECS	I/O'S USED	ELAPSED TIME (SECS)	CPU/E.T. (PER CENT)	JOB STEP COST
8:32:23	8:33:28	R4965#DI	1	168.	20.46	1294.	65.	31.48	9.96
8:32:36	8:57:26	Y6700#PT	1	52.	0.35	148.	1490.	0.02	0.25
8:33:29	8:33:32	R4965#UI	2	168.	0.0	74.	3.	0.0	0.37
8:33:33	8:33:36	R4965#DI	3	168.	0.0	62.	3.	0.0	0.31
8:38: 2	8:38:50	W2855#MG	1	104.	18.43	760.	48.	38.40	4.29
8:38:22	8:39: 1	O4600#1	1	104.	2.28	263.	39.	5.85	1.06
8:38:51	8:39:22	W2855#MG	2	104.	1.09	488.	31.	3.52	1.64
8:39: 2	8:39:32	O460C#1	2	104.	0.79	412.	30.	2.63	1.37
8:39:23	8:40:36	W2855#MG	3	104.	13.05	1609.	73.	17.88	6.38
8:39:33	8:39:47	O460C#1	3	168.	0.27	160.	14.	1.93	0.95
8:40:13	8:41:21	S817C#AA	1	104.	11.45	980.	68.	16.84	4.25
8:40:47	8:40:55	W2855#TR	1	54.	0.0	63.	8.	0.0	0.10
8:40:56	8:40:58	W2855#TR	2	54.	0.0	62.	2.	0.0	0.10
8:41:22	8:41:45	S8170#AA	2	104.	0.77	527.	23.	3.35	1.72
8:41:46	8:46: 6	S8170#AA	3	52.	4.28	4892.	260.	1.65	7.85
8:41:59	8:42:12	W1555#AL	1	104.	1.15	201.	13.	8.85	0.75
8:42:13	8:43: 3	W1555#AL	2	104.	2.73	849.	50.	5.46	2.93
8:43: 4	8:43:10	W1555#AL	3	52.	0.02	102.	6.	0.33	0.16
8:43:11	8:43:24	W1555#AL	4	104.	1.00	198.	13.	7.69	0.72
8:43:25	8:43:31	W1555#AL	5	52.	0.09	105.	6.	1.50	0.17
8:43:37	8:44:17	G6518#14	1	104.	2.75	477.	40.	6.87	1.77
8:44:18	8:44:31	G6518#14	2	104.	0.29	202.	13.	2.23	0.66
8:44:32	8:44:42	G6518#14	3	104.	0.17	172.	10.	1.70	0.55
8:44:46	8:45:20	F8285#CO	1	104.	3.32	512.	34.	9.76	1.94
8:46:37	8:48:15	H1200#TT	1	54.	0.75	625.	98.	0.77	1.05
8:49:11	8:49:39	M2900#0B	1	104.	5.83	354.	28.	20.82	1.71
8:49:41	8:50:42	M2900#0B	2	104.	2.42	802.	61.	3.97	2.75
8:50:44	8:50:57	M2900#0B	3	52.	0.04	102.	13.	0.31	0.16
8:50:58	8:51:25	M2900#0B	4	104.	1.95	433.	27.	7.22	1.55
8:51:26	9: 7:51	S0900#01	1	120.	205.64	17258.	985.	20.88	86.91
8:51:32	8:51:45	M2900#0B	5	52.	0.09	105.	13.	0.69	0.17
8:52:38	8:53:58	H1200#TT	1	54.	0.85	882.	80.	1.06	1.47
8:55:22	8:55:26	M9417#5	1	52.	0.01	80.	4.	0.25	0.13
8:55:27	8:55:37	M9417#5	2	52.	0.27	132.	10.	2.70	0.22
8:55:38	8:56:49	M9417#5	3	104.	16.48	1051.	71.	23.21	4.99
8:56:50	8:57:49	M9417#5	4	104.	1.89	686.	59.	3.20	2.34
8:57:29	9: 0:56	H1200#TT	1	54.	10.32	10566.	207.	4.99	17.67
8:59:31	9: 0:20	M9417#5	5	180.	1.02	239.	49.	2.08	1.47
8:59:38	8:59:59	B4562#33	1	104.	0.79	176.	21.	3.76	0.63
9: 0: 1	9: 0:45	B4562#33	2	104.	0.59	381.	44.	1.34	1.25
9: 1: 2	9: 1:37	B4562#33	3	52.	0.25	119.	35.	0.71	0.20
9: 1:13	9: 2:25	W317C#W1	1	104.	3.13	484.	72.	4.35	1.94
9: 2:26	9: 7: 9	G5010#RT	1	228.	6.00	911.	283.	2.12	7.60
9: 2:34	9: 3:36	W317C#W1	2	104.	0.40	204.	62.	0.65	0.68
9: 3:37	9: 4: 6	W3170#W1	3	52.	0.64	228.	29.	2.21	0.39
9: 6:34	9: 7:29	M7015#11	1	104.	6.99	485.	55.	12.71	2.24
9: 7:18	9: 7:46	F3840#JF	1	104.	1.25	213.	28.	4.46	0.79
9: 7:30	9: 8:38	W7888#9	1	52.	1.29	847.	68.	1.90	1.39
9: 7:40	9: 7:55	M7015#11	2	104.	0.0	74.	15.	0.0	0.23
9: 7:55	9: 8:46	F3840#JF	2	104.	0.65	387.	51.	1.27	1.28
9: 8: 9	9:23:14	B9700#LS	1	190.	0.22	135.	905.	0.02	0.81
9: 8:15	9:23:58	W7888#10	1	52.	0.0	161.	943.	0.0	0.25
9:23:53	9:27:15	F3840#JF	3	64.	23.20	260.	202.	11.49	1.94
9:24: 0	9:24: 6	B9700#LS	2	54.	0.0	51.	6.	0.0	0.09

Fig. 3--Job Step Accounting Report

The second program also generates a summary report of system performance and workload characteristics for a given time period. This report is illustrated in Fig. 4, which summarizes performance and workload for October 28, 1969. All terms used in the report are self-explanatory, with the possible exception of the last two items--"JOBS LOADED BY R.J.E." (Remote Job Entry jobs, which are batch jobs submitted from consoles remote from the computer room) and "C.P.S. JOBS" (Conversational Programming System jobs, which are on-line jobs submitted from remote consoles). Both items have zero jobs in the report because neither RJE nor CPS was operational until January 1970.

Because averages can often be deceiving, frequency distributions of some "workload characteristics" were thought helpful in getting a better idea of the workload processed. Therefore, the program also generates frequency distributions of the memory (core) requested per job step (Fig. 5), the CPU seconds used per job step (Fig. 6), and the Job I/Os used per job step (Fig. 7).

REQUIRED RESOURCES

The bulk of resources expended in reducing the accounting data was the human resources used in developing the data-reduction programs. The estimated resources spent (1) learning what accounting data were available and where and how they were stored and (2) writing, debugging, validating, and documenting the data-reduction programs was about four to six man-months for an experienced programmer-analyst. Also necessary (particularly in the early stages of development) were the consulting services of a systems programmer familiar with the operating system and the methods by which accounting data were recorded, collected, and stored.

The data-reduction process involved running five programs: two conditioning programs (one for the system log and one for the accounting log) and three processing programs (one for the system log and two for the accounting log). The conditioning program for the accounting log was merely a "card-to-tape" conversion routine supplied by the system. The other four programs were specially written to reduce the accounting data.

PERFORMANCE AND WORKLOAD SUMMARY REPORT
OCT 28, 1969

MONITORED INTERVAL

LENGTH OF MONITORED INTERVAL (IN HOURS) = 9.00
(ACTUAL CLOCK INTERVAL)
8:30 TO 17:30

TIME DURING WHICH COMPUTER WAS IDLE OR DOWN = 0.56

NET LENGTH OF MONITORED INTERVAL = 8.44

PERFORMANCE MEASURES

CPU UTILIZATION = 0.267

JOB I/O'S EXECUTED PER SECOND = 23.9

AVERAGE NUMBER OF JOBS ON THE COMPUTER = 2.70

NUMBER OF JOB STEPS PROCESSED = 646.
JOB STEPS PROCESSED PER HOUR = 76.5

NUMBER OF JOBS PROCESSED = 226.
JOBS PROCESSED PER HOUR = 26.8

TOTAL REVENUE = $3866.31
REVENUE PER HOUR = $ 458.09

WORKLOAD CHARACTERISTICS

AVERAGE I/O'S USED PER JOB STEP = 1126.

AVERAGE CPU SECONDS USED PER JOB STEP = 12.54

AVERAGE CPU SECONDS USED PER JOB STEP = 15.22
(EXCLUDING JOB STEPS WITH ZERO CPU SECONDS)

AVERAGE CORE REQUESTED PER JOB STEP = 97.

AVERAGE NUMBER OF JOB STEPS PER JOB = 2.9

JOBS RUN IN "STAND-ALONE" MODE = 0.

JOBS RUN IN "MULTI-PROGRAMMING" MODE = 226.
JOBS LOADED BY OPERATORS = 224.
JOBS LOADED BY R.J.E. = 0.
C.P.S. JOBS = 0.

Fig. 4--Performance and Workload Summary Report

FREQUENCY DISTRIBUTION OF CORE REQUESTED PER JOB STEP

DATE = OCT 28, 1969
TIME INTERVAL = 0830 - 1730

CORE INTERVALS	NUMBER OF JOB STEPS PER INTERVAL
0 - 50	0.
50 - 60	212.
60 - 70	29.
70 - 80	2.
80 - 90	2.
90 - 100	4.
100 - 110	301.
110 - 120	8.
120 - 130	0.
130 - 140	1.
140 - 150	11.
150 - 160	4.
160 - 170	28.
170 - 180	10.
180 - 190	3.
190 - 200	3.
200 - 225	7.
225 - 250	21.
250 - 275	0.
275 - 300	0.
OVER 300	0.

Fig. 5--Core Requested per Job Step

FREQUENCY DISTRIBUTION OF CPU SECONDS USED PER JOB STEP

DATE = OCT 28, 1969
TIME INTERVAL = 0830 - 1730

CPU SECOND INTERVALS	NUMBER OF JOB STEPS PER INTERVAL	CPU SECONDS USED	PER CENT OF TOTAL CPU SECONDS USED
0	114.	0.	0.0
0 - 10	387.	746.	9.2
10 - 20	58.	853.	10.5
20 - 30	27.	678.	8.4
30 - 40	13.	454.	5.6
40 - 50	9.	405.	5.0
50 - 60	6.	328.	4.1
60 - 70	2.	128.	1.6
70 - 80	4.	302.	3.7
80 - 90	3.	248.	3.1
90 - 100	2.	195.	2.4
100 - 200	16.	2393.	29.5
200 - 300	4.	948.	11.7
300 - 400	0.	0.	0.0
400 - 500	1.	420.	5.2
500 - 600	0.	0.	0.0
600 - 700	0.	0.	0.0
700 - 800	0.	0.	0.0
800 - 900	0.	0.	0.0
OVER 900	0.	0.	0.0

Fig. 6--CPU Seconds Used per Job Step

FREQUENCY DISTRIBUTION OF I/O'S USED PER JOB STEP

DATE = OCT 28, 1969
TIME INTERVAL = 0830 - 1730

I/O INTERVALS	NUMBER OF JOB STEPS PER INTERVAL	I/O'S USED	PER CENT OF TOTAL I/O'S USED
0 - 100	120.	7799.	1.1
100 - 200	81.	12549.	1.7
200 - 300	62.	14569.	2.0
300 - 400	65.	23151.	3.2
400 - 500	65.	29025.	4.0
500 - 600	45.	24561.	3.4
600 - 700	12.	7773.	1.1
700 - 800	15.	11309.	1.6
800 - 900	18.	15290.	2.1
900 - 1,000	23.	21787.	3.0
1,000 - 2,000	61.	90241.	12.4
2,000 - 3,000	17.	42386.	5.8
3,000 - 4,000	18.	64541.	8.9
4,000 - 5,000	11.	49815.	6.8
5,000 - 6,000	2.	10013.	1.4
6,000 - 7,000	8.	52341.	7.2
7,000 - 8,000	5.	38649.	5.3
8,000 - 9,000	5.	42151.	5.8
9,000 - 10,000	1.	9108.	1.3
10,000 - 11,000	4.	41768.	5.7
11,000 - 12,000	2.	22293.	3.1
12,000 - 13,000	1.	12480.	1.7
13,000 - 14,000	0.	0.	0.0
14,000 - 15,000	2.	29244.	4.0
15,000 - 16,000	0.	0.	0.0
16,000 - 17,000	0.	0.	0.0
17,000 - 18,000	1.	17258.	2.4
18,000 - 19,000	2.	37405.	5.1
19,000 - 20,000	0.	0.	0.0
20,000 - 30,000	0.	0.	0.0
30,000 - 40,000	0.	0.	0.0
40,000 - 50,000	0.	0.	0.0
OVER 50,000	0.	0.	0.0

Fig. 7--I/Os Used per Job Step

The programs were written so that different options regarding the characteristics of input data and output reports could easily be specified through the use of input control parameters. Thus, the programs were written in the form of production codes in that (1) they fully documented what functions the various segments of the programs performed, and (2) they provided the user with instructions for submitting the programs.

Extra effort was expended writing production codes instead of "quick and dirty" programs because (1) we intended to use the programs continuously and did not want to go through the process of learning what the programs did and how to submit them each time we used them, and (2) the programs might be used to reduce accounting data from computer systems similar to Rand's (e.g., Air Force or NASA computer installations equipped with IBM 360 Series computer systems).

III. APPLICATIONS OF ACCOUNTING DATA

INTRODUCTION

After the accounting data have been conditioned and reduced, they can be used to measure and evaluate system performance in a manner similar to data from system monitors or system models.

The applications discussed below utilize the reduced data discussed in Sec. II, particularly the "workload characteristics" and "performance measures" presented in Fig. 5. In this report, workload characteristics are the computer resources used by the job steps processed by the system (averaged for all job steps processed during a given time period). Performance measures, in this report, refer to *efficiency* when measuring the performance of the CPU (i.e., CPU utilization), and to *throughput* for all other performance measures (i.e., job steps processed per hour, etc.).

A number of applications for these data were discovered in performance studies of Rand's computer system. Some of the applications utilize accounting data as the sole source of measurement data. Others use accounting data in conjunction with either system-monitoring devices (hardware or software) or system models (simulation or analytical). The following applications were found to be most useful:

1. Measuring and evaluating the effects of a system modification.
2. Use in conjunction with hardware or software monitors to measure and improve computer system performance.
3. Formulating or revising computer charging schemes.
4. Providing background information for a performance improvement effort.
5. Providing installation management with up-to-date reports on computer usage and workload characteristics.
6. Providing measurements and descriptions of workloads processed by the system in the design of "typical" job streams (to be used in making benchmark tests of alternative computer systems, or as input data for simulation or analytical models).
7. Compiling trends of past usage and performance (to be used in forecasting future demands on the system).

Most of the research effort was directed at the first four applications because these were considered to be most useful in measuring and evaluating computer system performance. Moreover, of these four, most of the time and effort was spent measuring and evaluating the effects of a system modification because this application was considered to be most valuable.

This section discusses the above applications according to the amount of research devoted to each. Therefore, the primary focus is on the first application; the remainder of the discussion centers on the second, third, and fourth applications.

MEASURING THE EFFECT OF A SYSTEM MODIFICATION

We found that accounting data can be very useful in measuring the effect of a system modification on the performance of a computer system. In using them for this application, we analyze performance from many days of accounting data before and after the modification and then measure changes in various performance measures to determine the effect of the modification.

The use of accounting data for this application has distinct advantages and disadvantages compared to the widely used method of running controlled tests on the system before and after the modification and measuring performance changes with hardware or software monitors.

The advantages of using accounting data include:

1. The rental or procurement costs of sophisticated hardware or software monitors are not incurred.
2. The system's operation is not disrupted, as it would be if controlled tests were run.
3. There is no need to design a "typical" job stream to use in running controlled tests.
4. Workload characteristics for the jobs processed are provided; these are not provided by hardware or software monitors.
5. Accounting data are cheap to obtain because they are automatically collected by most third-generation computer systems.

6. Accounting data can be used to measure the effects of a system modification made some time in the past merely by analyzing the accounting data collected at that time; using hardware or software monitors to measure the effects of a past modification requires considerable effort to duplicate the workload processed and the hardware and software configurations that existed at that time.

The use of accounting data to measure the effect of a system modification has the following disadvantages:

1. Accounting data require considerably more conditioning and reducing than do data from hardware or software monitors.
2. Accounting data are not as accurate as the data provided by software or hardware monitors.
3. Accounting data do not measure the usage or utilization of as many system components as do hardware or software monitors.
4. The modification to be evaluated must have been implemented for a sufficient amount of time so that enough accounting data have been collected to perform valid analyses.

The use of accounting data to measure the effect of a system modification is illustrated by measuring the effect of a recent system modification at Rand. The modification was the addition of 256K bytes of high-speed core memory to our IBM 360/65 computer system, which previously had 512K bytes of high-speed core memory. Actually, the net amount of memory available for application programs after the modification was about 520K bytes because of the memory requirements of the operating system and other system routines that were permanently stored in core memory. Since the net amount of memory available before the modification was about 320K bytes (out of 512K bytes), the modification resulted in a net increase of about 200K bytes of memory. Figure 8 diagrams the computer-system configuration after the modification.

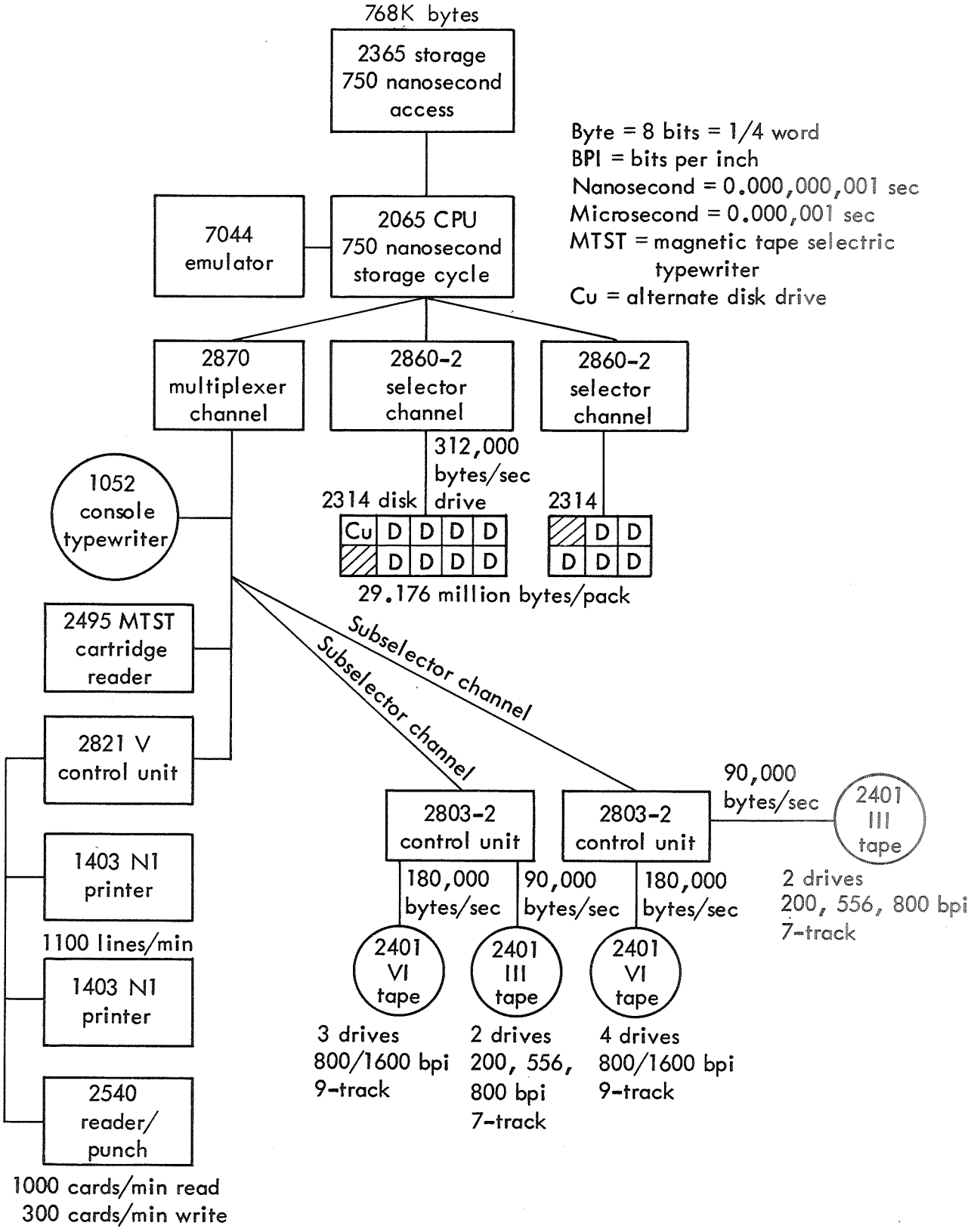

Fig. 8--System Configuration for Rand's IBM 360/65 Computer System

Developing a Methodology

In practice, it was much more difficult than expected to measure the effects of a system modification by analyzing the accounting data before and after the modification. We expected a situation as in Fig. 9, where values for various performance measures could be computed for a number of days before and after the modification and the change in performance determined by comparing the respective performance levels. Such a situation actually did exist in measuring the effect of the modification upon the average number of job steps in core. Figure 10 plots the average number of job steps in core before and after the core modification and indicates that the level increased by approximately one job step after the modification. However, the average number of job steps in core was not considered a performance measure since it neither measured system throughput nor system efficiency. In contrast, plots of such performance measures as CPU utilization, job I/Os processed per second, job steps processed per hour, etc. resembled scatter diagrams more than the expected step function depicted in Fig. 9. Figures 11 and 12 give examples of actual graphs; there is no indication that the modification had any effect upon these performance measures.

At this point in our analysis, we did not know whether the modification had little or no effect upon system performance or whether it actually significantly increased (or decreased) system performance but our method of analysis was inadequate to detect such changes.

We tried using more days of data in the analysis--in fact, we used as many days before and after the modification as possible. It turned out that an eight-drive disk storage module (an IBM 2314) had been added to the system about a month and a half before the modification; the Christmas season began about two months after the modification. Therefore, we were restricted to using only data between these two events. Eliminating holidays, weekends, and days when the computer system was down left 34 days of data before the modification and 33 days of data after.

Only data generated during the busy portion of each day were used (8:30 a.m.-5:30 p.m. at Rand) because the measures used in analyzing

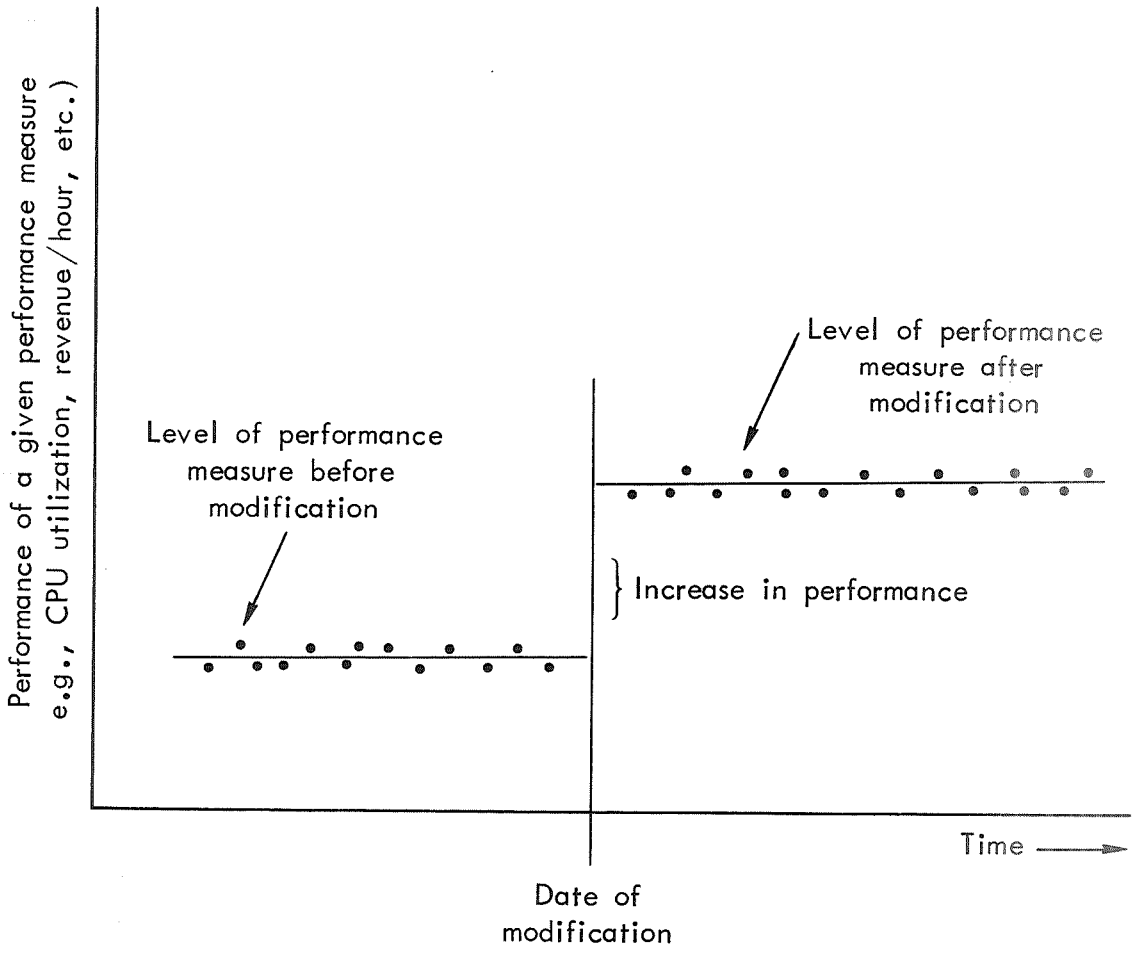

Fig. 9--Expected Comparison of Performance Before and After the Modification

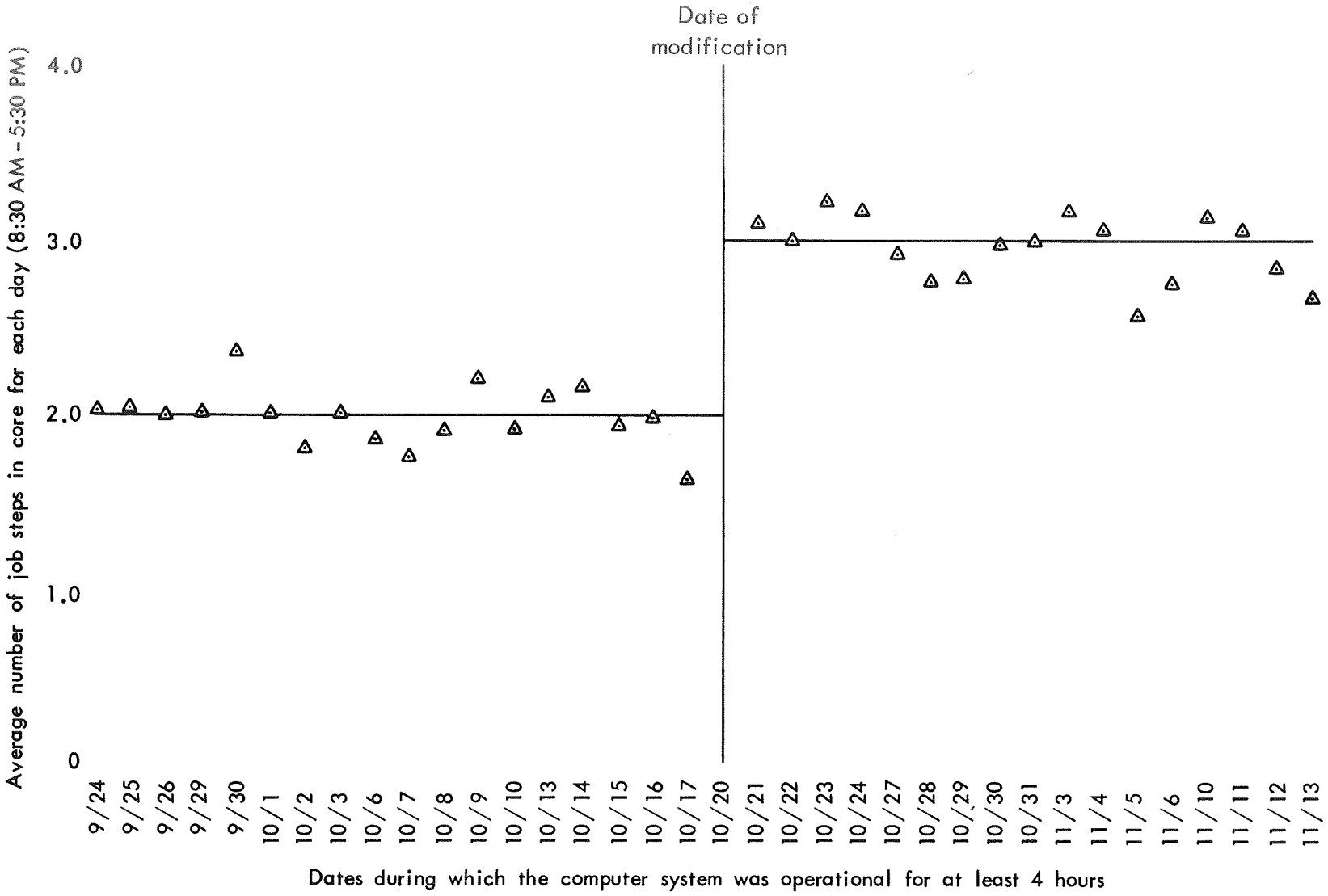

Fig. 10--Jobs in (Core) Memory Before and After 256K of High-Speed Core Were Added to the System

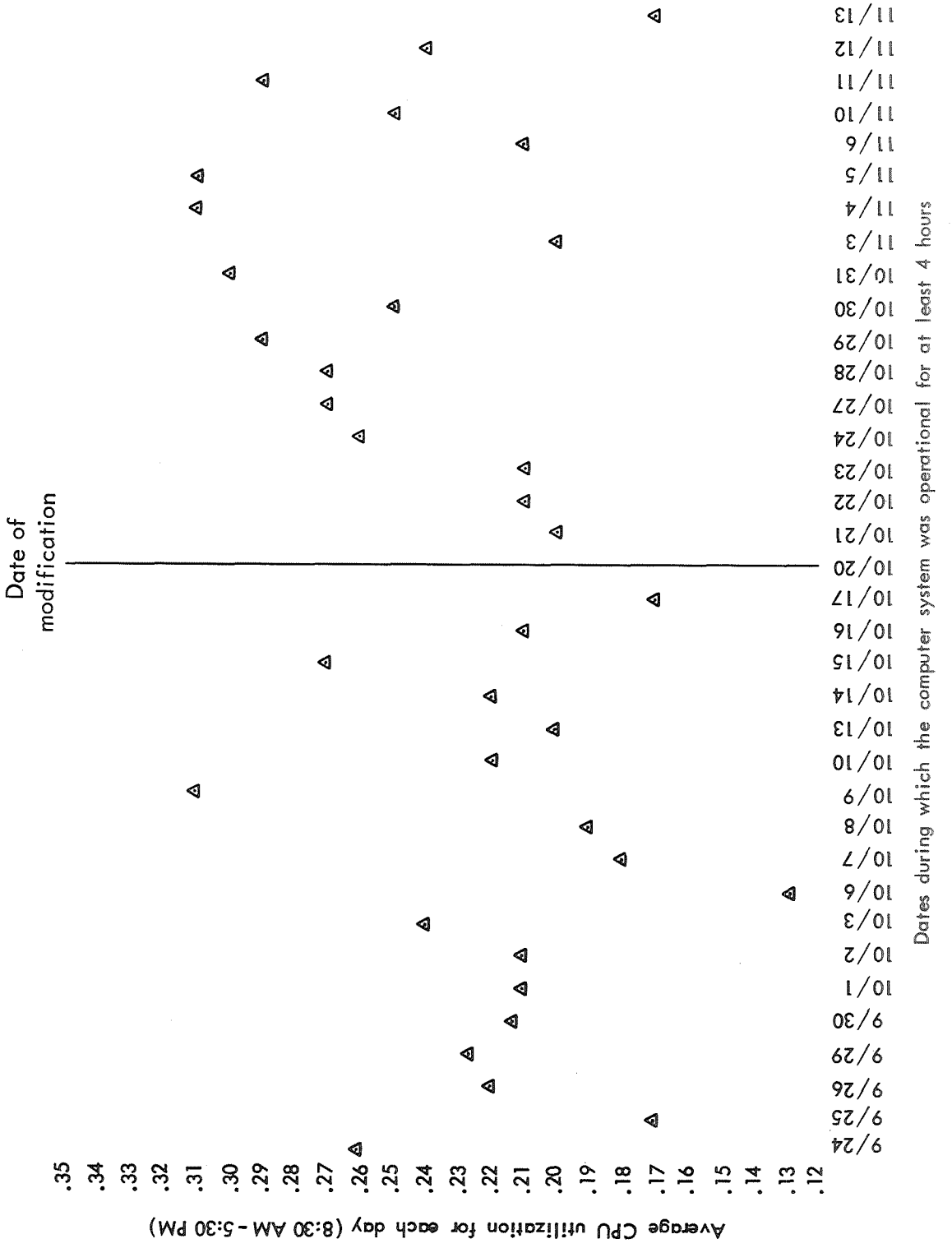

Fig. 11--CPU Utilization Before and After Core Modification

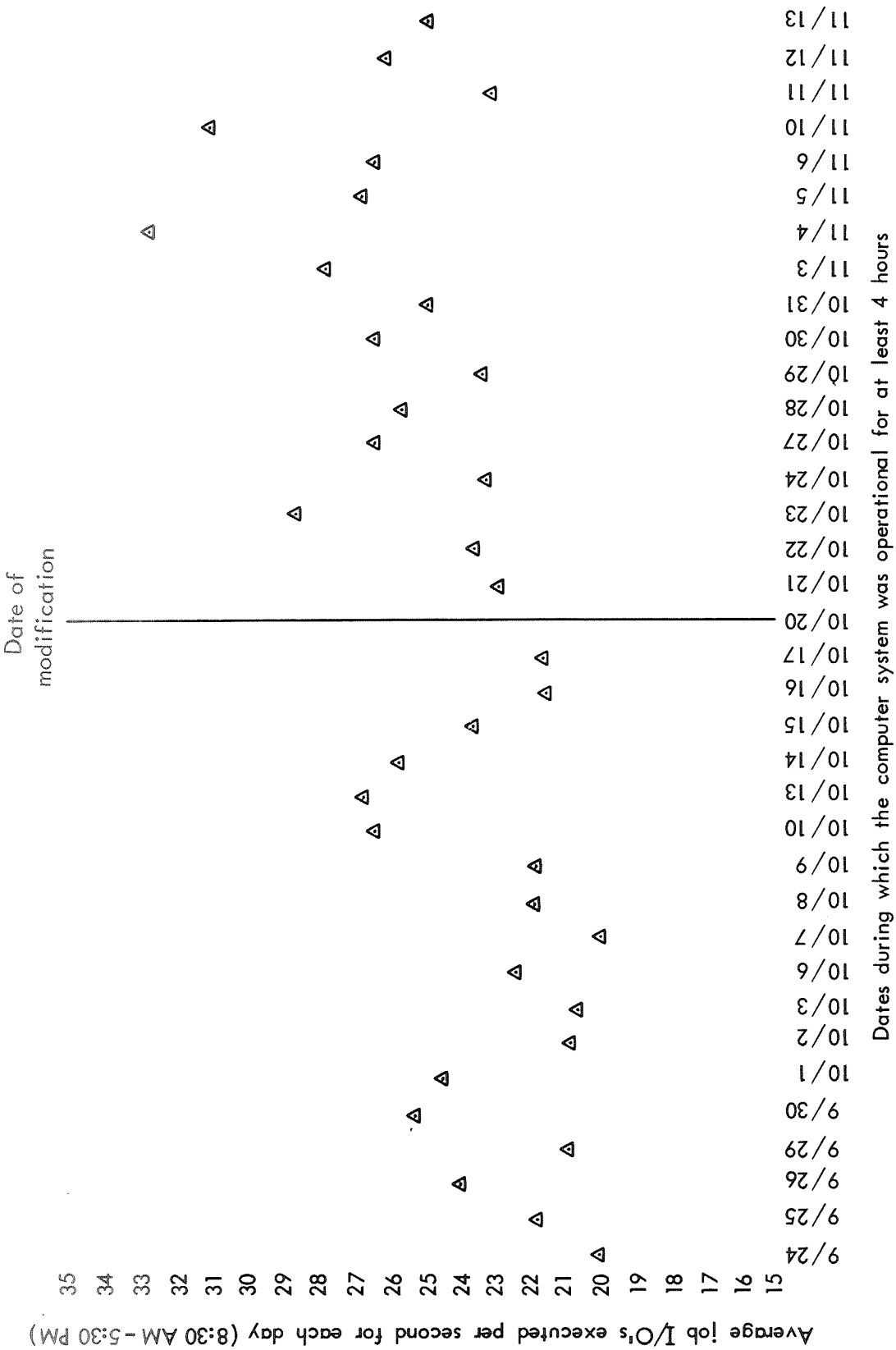

Fig. 12--Average Job I/Os Executed per Second Before and After the Core Modification

performance (i.e., CPU utilization, I/Os processed per second, etc.) were primarily measures of system throughput and as such had little validity during nonbusy hours. However, even during the busy hours our analyses indicated that performance and workload varied considerably;[†] therefore, we divided each day into three independent 3-hr time periods (8:30-11:30, 11:30-2:30, and 2:30-5:30). Average values for various performance measures and workload characteristics were then calculated for each time period.[‡] After eliminating all periods in which the computer was either down or idle for more than 1 hr during each 3-hr period, the total number of periods left was 81 (before) and 78 (after).

The analysis was then redone, using these additional data and average performance measures for 3-hr periods instead of 9-hr days. Unfortunately, we got the same results as in the previous analysis--no indication whatsoever of an increase or decrease in performance.

Had we not calculated and plotted several workload characteristics in addition to calculating and plotting various performance measures, we probably would have concluded that accounting data were too variable to be used in performance analysis--and discontinued the effort. However, we noticed striking similarities between plots of various workload characteristics and performance measures. Therefore, we ran the data through correlation- and regression-analysis programs to determine

---

[†]Figures 5, 6, and 7 and the graphs in the Appendix illustrate this variability.

[‡]Naturally, since average values were used to represent the workload characteristics and performance measures over each 3-hr period, the use of shorter time periods would have increased the validity of these average values. However, overlapping jobs (due to the multiprogramming environment) result in a number of job steps starting in one time period and terminating in the next consecutive time period. Since the resources used by each job step are recorded when the job step terminates, errors in allocating resources to the proper time periods are introduced because there is no way of knowing which quantities of resources are used before one time period ends and the next one begins. Thus, although we would have preferred to use time periods of less than 3 hr, in shorter time periods these "overlapping" errors were too great with respect to the total resources used.

how close the relationships were. We found that average CPU seconds used per job step were very highly correlated with CPU utilization. This is illustrated in Fig. 13, where linear regression analysis is used to estimate CPU utilization from average CPU seconds used per job step. Figure 14 illustrates the high degree of correlation between average I/Os used per job step and average job steps processed per hour.

These correlation and regression analyses showed that the fluctuations in performance were caused almost completely by fluctuations in the characteristics of the workloads processed. We also learned that if accounting data were to be used to measure the effects of a system modification, the influence of workload characteristics upon performance had to be either eliminated or significantly reduced.

In subsequent studies, we developed two independent methods of analyzing the accounting data--both of which overcame the problem of comparing performance measures when different workloads were processed by the system. Both methods involve statistical analysis of the data. The first uses multiple regression analysis; the second utilizes cluster analysis.

Regression Analysis Method

Basically, the regression analysis method uses a linear multiple regression of workload characteristics to estimate each performance measure. A modification variable was added to the workload characteristics and assigned a value of 0 before the modification and 1 after the modification. Therefore, the value of the regression coefficient calculated for this variable was the estimated change in the particular performance measure resulting from the modification. The following discussion details this method.

The *workload characteristics* (treated as independent variables in the regression analysis) were

x_1 = Average number of I/Os used per job step.

x_2 = Average CPU seconds used per job step.

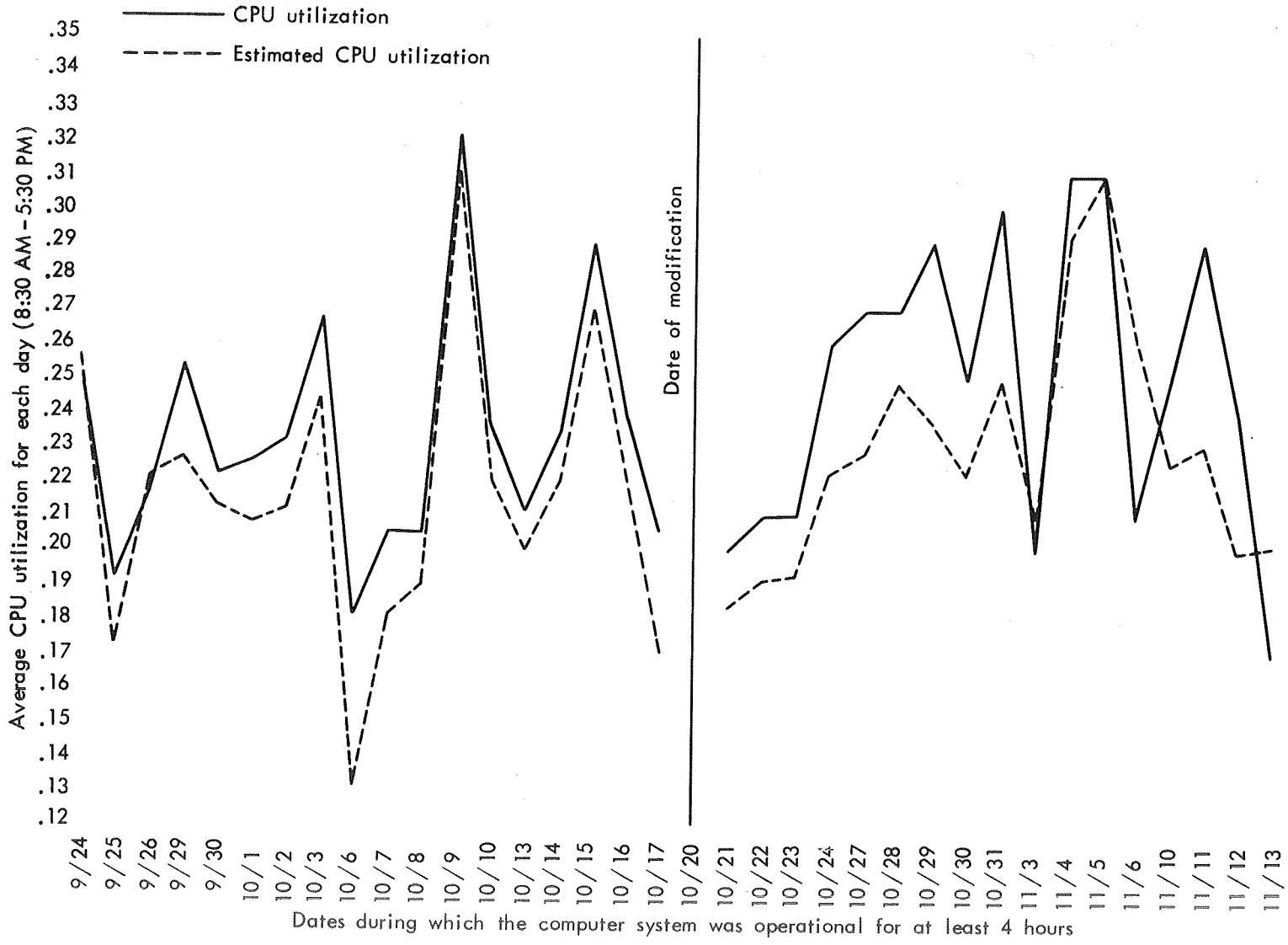

Fig. 13--CPU Utilization and Estimated CPU Utilization from a Linear Regression of Average CPU Seconds Used per Job Step

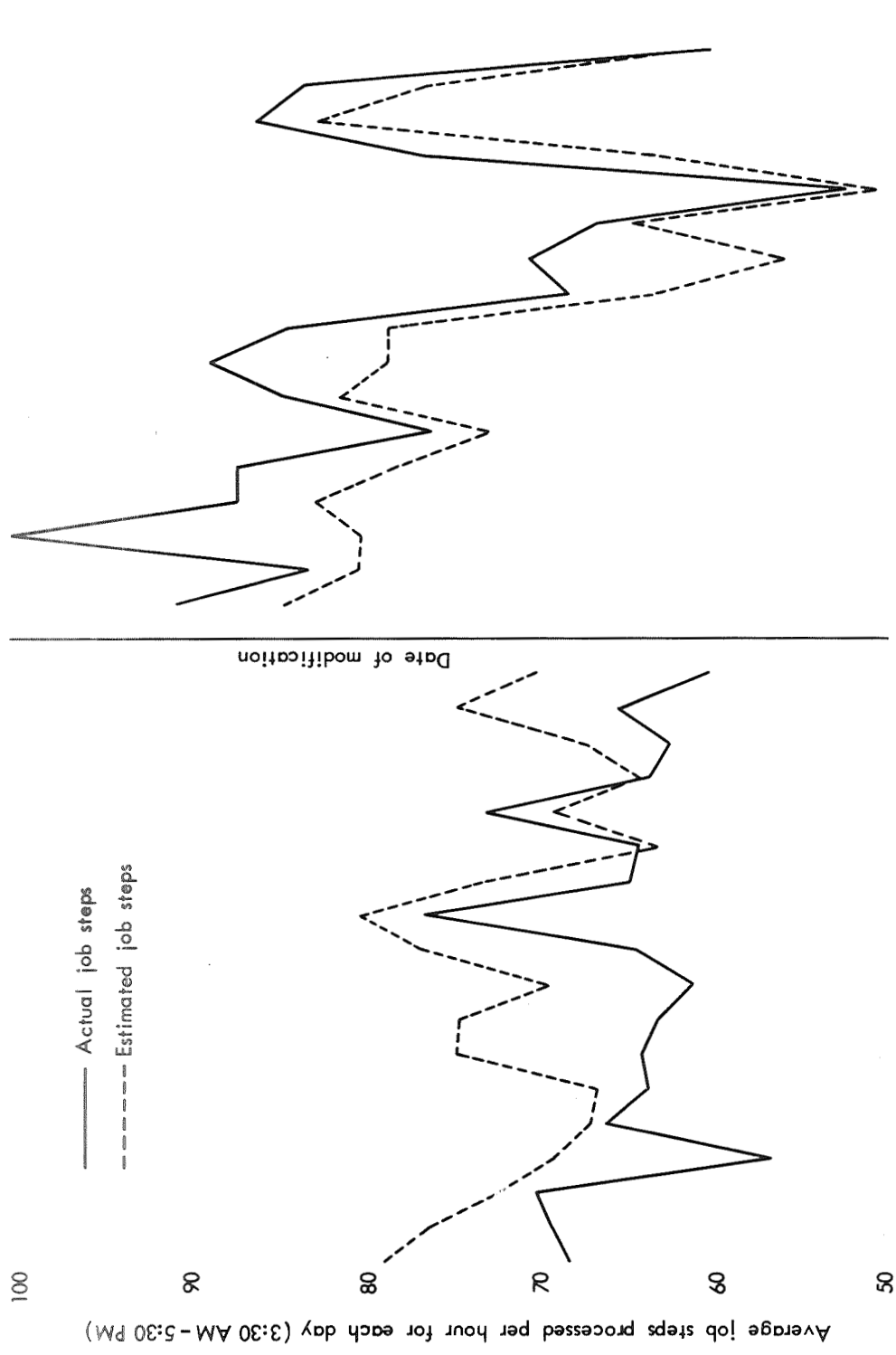

Fig. 14--Actual Job Steps Processed per Hour and Job Steps Processed per Hour Estimated from a Linear Regression of Average I/Os Used per Job Step

x_3 = Average CPU seconds used per job step, excluding job steps using 0 CPU seconds (i.e., excluding those job steps not actually running).†

x_4 = Average core requested per job step.

x_5 = Average number of job steps per job.

x_6 = Trend variable (i.e., the time within a 3-1/2 month period in which the 3-hr time periods were extracted).

x_6 = 1, ..., 81 (sequentially on time of day and date for each time period before the modification).

x_6 = 82, ..., 159 (sequentially on time of day and date for each time period after the modification).

x_7 = Modification variable,

x_7 = 0 before the modification,

x_7 = 1 after the modification.

The *performance measures* (treated as dependent variables in the regression analysis) were

x_8 = Average number of job steps in core.

x_9 = Average CPU utilization.

x_{10} = Average I/Os processed per second.

x_{11} = Average job steps processed per hour.

x_{12} = Average jobs processed per hour.

x_{13} = Average revenue produced per hour.

It was not clear whether variable x_8 should be included as a performance measure. The modification affected this variable more than any of the performance variables, increasing its average value from 1.99 jobs in core before the modification to 3.01 jobs in core after the modification. However, such an increase was expected because the modification involved increasing the computer system's

---

†Often, an error in a previous job step within a job cancels the job from further execution. Therefore, all subsequent job steps in such a job do not use the CPU (they use zero CPU seconds). However, these job steps must be considered in the analysis because they are still loaded on and off the computer system and, in the process, use I/O resources.

core storage. Therefore, since this variable was more a measure of system capacity than a measure of system throughput or efficienty, it was excluded from the analysis.

The regression equation took the following form:

$$y = \hat{A}_0 + \hat{A}_1 x_1 + \hat{A}_2 x_2 + \hat{A}_3 x_3 + \hat{A}_4 x_4 + \hat{A}_5 x_5 + \hat{A}_6 x_6 + \hat{A}_7 x_7 \qquad (1)$$

where y represents the conditional mean of the dependent variable (taken to be x_9, x_{10}, x_{11}, x_{12}, x_{13} at different times) as a function of the workload characteristics x_1, x_2, x_3, x_4, x_5, x_6, x_7. $\hat{A}_0$ is the intercept, and the other $\{\hat{A}_i\}$ (i=1-7) are the regression coefficients for their respective workload characteristics, $\hat{A}_i$ being the conditional regression coefficient between y and x_i when the other x_j, $j \neq i$ are held fixed (as in a laboratory). $\hat{A}_i$ is the sample estimate of the true but unknown coefficient A_i.

Since $x_7 = 0$ before the modification and $x_7 = 1$ after the modification, then for any fixed $\{x_1, x_2, x_3, x_4, x_5, x_6\}$ workload characteristics,

$$y_{before} = \hat{A}_0 + \sum_{i=1}^{6} \hat{A}_i x_i \qquad (\text{Since } \hat{A}_7 x_7 = 0) \qquad (2)$$

$$y_{after} = \hat{A}_0 + \sum_{i=1}^{6} \hat{A}_i x_i + \hat{A}_7 \qquad (\text{Since } \hat{A}_7 x_7 = \hat{A}_7) \qquad (3)$$

represent the expected performance before and after the modification.

Since x_7 is 0 before the modification and 1 after the modification,

$$\hat{\Delta} = y_{after} - y_{before} = \hat{A}_7 \qquad (4)$$

is the estimated change in performance for the given workload.

Thus, even though the workload was not the same for the two periods, the major portion of the difference has been accounted for by the six workload variables used. Thus, the measured difference $\hat{\Delta}$ reflects

the change in system performance resulting from the modification and not the change created by different workload conditions (unless some other workload characteristic besides $x_1 - x_6$ was critical, which is unlikely).

To consider a special case, let $y = x_9$ (CPU utilization). The estimated regression coefficients $\hat{A}_i$ were calculated, to give

$$x_9 = 0.17631 - 0.00007\ x_1 + 0.01942\ x_2 - 0.00158\ x_3 - 0.00104\ x_4 + 0.01321\ x_5 - 0.00010\ x_6 + 0.03495\ x_7 \quad (5)$$

with a multiple correlation of

$$R = 0.90257 \quad (6)$$

The estimated mean change in CPU utilization (resulting from the modification) was the calculated value of $\hat{A}_7$, or + 0.03495.

The effect of the trend variable x_{12} tended to be small for all performance characteristics and was negative for all performance characteristics other than x_{11} (revenue). Therefore, had the trend variable been ignored, conclusions concerning the modificatin would have been substantially the same although the improvements in Δ would have appeared to be slightly less for performance characteristics other than x_{11} (revenue) and slightly greater for variable x_{11}.

The results of the regression analysis were primarily indicated by the values of A_7 (regression coefficient for the modification variable) in the regression equations for the various performance variables. Discussed in context with the average values for these variables before the modification, the modification resulted in a fractional increase in performance of about 17 percent for the 5 performance measures evaluated. However, there is some doubt as to the size of the increase for each variable when the change in performance is expressed in terms of confidence intervals (see Table 1).

Table 1

EFFECTS OF THE MODIFICATION AS MEASURED BY REGRESSION ANALYSIS

Performance Variable	Mean Before	$\hat{\Delta}$	Fract. Increase	95% Conf. Int. for Δ
CPU utilization (X_9)	.218	+.0350	.161	(.0190 - .0510)
I/Os processed/ second (X_{10})	23.09	+3.27	.142	(1.54 - 4.99)
Job steps processed/ hour (X_{11})	69.58	+12.38	.178	(7.36 - 17.41)
Jobs processed/hour (X_{12})	25.61	+4.67	.182	(2.83 - 6.51)
Revenue (dollars) produced/hour (X_{13})	358.60	+59.82	.167	(33.03 - 86.61)

Cluster Analysis Method

In this method, clusters of time periods were selected, each cluster containing time periods in which similar workloads were processed. For all clusters that contained at least one time period before and one after the date of the modification, the increase (or decrease) in performance that resulted from the modification could be calculated. The performance changes for all such clusters were statistically analyzed to determine confidence intervals of improvement (or degradation) resulting from the modification.

The cluster analysis method is more "robust" than the regression analysis method in that it gives a good analysis over a wider class of assumptions. Therefore, the relative merits of the two methods depend on the appropriateness of the regression model. If the assumptions[†]

[†]The assumptions for using the regression analysis method were (1) linear relationships between the independent and dependent variables, (2) additivity among the independent variables in estimating dependent variables, and (3) equal variances for the dependent variables.

of the regression analysis method are appropriate, it gives better results because (1) it uses all the data, and (2) it gives smaller confidence intervals because it matches the workload characteristics in an optimal, continuous manner. On the other hand, cluster analysis gives better results when the assumptions of regression analysis are invalid.

The cluster analysis method is a little more difficult to apply because it requires an initial analysis to form clusters of the data (time periods) and then an analysis of the clusters. The regression analysis method only requires analysis of the data (time periods).

The use of a cluster analysis method of measuring changes in performance may not require developing a special method for forming clusters. In fact, if cluster analysis programs are available at a computer installation (usually in the form of a package of statistical analysis programs), their use may result in substantial savings in time and effort. Although such a program was available at Rand, the fact that we knew a great deal about how the workload characteristics affected the various performance measures influenced us in developing our own method.

Developing a Method to Form Clusters. The principal problem associated with the cluster method was forming clusters of time periods that processed similar workloads--and doing this from the limited amount of data available. These data consisted of the workload characteristics for each 3-hr time period (the same workload data that were used in the regression analysis method), plus frequency distributions for each 3-hr period of (1) core requested per job step, (2) CPU seconds used per job step, and (3) I/Os used per job step.

We found that performance measures were very useful in developing a method for forming clusters because they could be used to test the similarity of the workloads: processing similar workloads results in similar performance--given that no system modifications have been made and no trends exist that affect system performance. The only restriction in using performance measures to test the similarity of the workloads was that all time periods in each test cluster had to be either before or after the modification.

The development of a cluster selection method consisted primarily of trial and error studies in which we utilized many combinations of the available workload information. Although at first we thought that the frequency distributions of the workload characteristics would be of most use as indicators of the workload processed during a time period, they proved to be very inconsistent in predicting performance. Eventually, they were discarded in favor of using only average workload characteristics. Linear correlation analyses of workload variables versus each of the performance variables were also very useful in developing a method for forming clusters. From these analyses (summarized in Table 2), we could see which workload variables had the most effect upon each performance variable. The analyses were used to attach to each of the workload variables a range of values within which the corresponding workload characteristics for each time period in a cluster were allowed to vary. These ranges were then used as a set of requirements that two or more time periods had to satisfy in order to form a cluster.

These ranges were initially set very narrowly for the workload variables we felt to be most important (average I/Os per job step, average CPU seconds used per job step, and average CPU seconds used per job step excluding jobs using zero CPU seconds), and set at broad values for the less-important workload variables (average core requested per job step, and average job steps per job). For convenience in trying out many different ranges for each workload variable, the ranges for each workload variable were set as multiples (or fractions) of their respective standard deviations.

In experimenting with various combinations of "range of values," we aimed at achieving a balance--on the one hand, ranges set narrow enough so that the time periods within each cluster processed very similar workloads and, on the other hand, ranges set broad enough so that a good portion of the time periods could be fitted into clusters.

The cluster selection method that evolved is described below in terms of (1) a range of values for each workload characteristic (Table 3), and (2) restrictions that apply when using these ranges to form clusters.

Table 2

LINEAR CORRELATION COEFFICIENTS BETWEEN WORKLOAD CHARACTERISTICS AND PERFORMANCE MEASURES (BEFORE/AFTER THE MODIFICATION)

Workload Characteristics	Performance Measures				
	CPU Utilization	Avg. I/Os Processed/Sec	Avg. Job Steps Processed/Sec	Avg. Jobs Processed/Hr	Avg. Revenue Produced/Hr
Avg. I/Os used per job step	-.17/.02	.65/.66	-.60/-.76	-.47/-.52	.18/.24
Avg. CPU seconds used per job step	.84/.77	-.17/.16	-.44/-.58	-.41/-.41	.55/.54
Avg. CPU seconds used per job step (minus 0-sec jobs)	.81/.79	-.20/.16	-.44/-.48	-.41/-.34	.52/.55
Avg. core requested per job step	.26/.38	-.33/-.03	-.38/-.48	-.29/-.31	.51/.45
Avg. job steps per job	.16/-.06	.03/-.23	.43/.44	-.18/-.23	.09/-.24
Multiple correlation of all workload characteristics	.92/.90	.79/.72	.78/.82	.74/.79	-.69/.59

Table 3

RANGES OF VALUES FOR WORKLOAD CHARACTERISTICS

Workload Characteristics	Range of Values[a]
Average I/Os used per job step	1/4 SD[b] (66 I/Os)
Average CPU seconds used per job step	1/4 SD (0.76 CPU sec)
Average CPU seconds used per job step (excluding job steps using 0 CPU sec)	1/4 SD (0.87 CPU sec)
Average core requested per job step	2 SD (13.8K bytes)
Average job steps per job	2 SD (0.60 job steps/job)

[a]Range of values within which the corresponding workload characteristics for each time period in a cluster can vary.

[b]Standard deviation.

In using Table 3 to form clusters, the following restrictions apply:

1. No time period may appear in more than one cluster.
2. If a time period can fit in more than one cluster, it is assigned to the cluster with the fewest number of time periods on the same side of the modification as the time period in question. (Although this restriction appears arbitrary, the ranges were set narrow enough so that it was only used two or three times.)

Results of the Cluster Analysis. Once the clusters were formed, it was much easier to analyze them for performance changes resulting from the modification. For each performance measure, the estimated change in performance (before versus after the modification) was calculated as a weighted average to account for unequal cluster sizes and to give more weight to clusters with more time periods. The standard deviation of the change in performance was a measure of the randomness caused by workload fluctuation. This randomness resulted not only

because the time periods in each cluster (though similar) were not identical, but also because performance changes were measured for different levels of workloads (each cluster processing a different workload). Finally, the t-distribution was used to calculate confidence intervals.

Table 4 gives results of the cluster analysis.

Table 4

EFFECTS OF THE MODIFICATION AS MEASURED BY CLUSTER ANALYSIS

Performance Measure	Mean Before	$\hat{\Delta}$	Fract. Increase	95% Confidence Interval for Δ
CPU utilization	.218	+.038	.174	.030 - .046
I/Os processed/second	23.09	+3.91	.169	3.01 - 4.84
Job steps processed/hour	69.58	+13.68	.197	10.05 - 17.29
Jobs processed/hour	25.61	+3.75	.150	3.25 - 4.25
Revenue ($) produced/hour	358.60	+70.14	.196	45.69 - 94.59

Comparison of Results of the Two Methods

In comparing results of the two methods, the confidence intervals of change from the cluster analysis method were *much smaller* than those from the regression analysis method. This does not mean that the cluster analysis method is a better method; it does mean that cluster analysis was better for the data used. For different data, regression analysis may be a better method.

Another unexpected difference in results between the two methods was that cluster analysis measured a slightly higher increase in overall performance than regression analysis. In re-evaluating the two methods, we found that this was because (1) Rand's computer system was

essentially an I/O-bound system, and (2) the cluster method used very few time periods with high average I/Os used per job step.†

Because the system was I/O bound, the core added by the modification had little effect upon performance when the system processed heavily I/O-oriented jobs. Therefore, the cluster method indicated higher increases in performance than the regression method, which used all the time periods in its analysis.

Interpretation of Results from this Application of Accounting Data

The regression analysis and cluster analysis methods illustrate how accounting data can be used to measure the effect of a system modification. In this particular case, we measured the effect of the modification by measuring the changes in 5 indicators of system throughput and efficiency. All 5 showed an increase of between 15 and 20 percent over their respective performance levels before the modification.

Note that this analysis provides information about increases (or decreases) in system performance attributable to the modification; it does not evaluate the modification in terms of comparing it with alternative modifications, analyzing cost-effectiveness, etc. Installation management must use this information to evaluate the modification with respect to such factors as the cost of the modification, the objectives of the computer installation (profit-making, cost-supported, overhead-supported), and potential increases in programmer productivity through faster job turnaround.

USE OF ACCOUNTING DATA IN CONJUNCTION WITH HARDWARE OR SOFTWARE MONITORS

With the increasing popularity of hardware and software monitoring devices to measure computer system performance, the use of accounting

---

†Very few of these time periods had sufficiently similar workload characteristics to be fitted into clusters; none of the highest 20 time periods could be fitted into clusters and only 7 of the next 20 were fitted.

data has taken on new roles. If a monitoring device is periodically used to measure performance, accounting data are very useful in determining the typicalness of the workload processed during the monitored time interval. Also, comparing performance measures obtained from hardware or software monitors to those calculated from accounting data is useful in determining how much system resources are consumed by unrecorded system overhead.

Determining the Typicalness of the Workload

Accounting data can be used to verify that a typical workload of jobs was run during a monitored period. This is determined by comparing workload and performance characteristics during the monitored period with workload and performance characteristics during previously monitored periods.

A scan of the computer resources used by each job step (as reported in Fig. 3, p. 17) is useful in:

1. Checking the CPU-boundness of jobs by observing (a) the number of CPU seconds used by each job step, and (b) the ratio of CPU seconds to total time on the computer.
2. Checking the I/O-boundness of jobs by observing the number of I/Os used by each job step.
3. Checking to see if a job unusually dominated the computer system over the monitored interval.

Summary figures for the monitored interval (as reported in Fig. 4, p. 19) are useful in comparing the workload and performance characteristics with previously monitored periods. Performance measures that proved useful at Rand include:

- Average CPU utilization.
- Average job I/Os processed per second.
- Average job steps processed per hour.
- Average number of jobs in core.
- Average revenue produced per hour.

Workload measures that proved useful at Rand include:

- Mean and frequency distribution of CPU seconds used per job step.
- Mean and frequency distribution of I/Os used per job step.
- Mean and frequency distribution of core memory requested per job step.

With these simple measures, we often determined that impressive results were due to shifts in job-stream characteristics rather than improvements in the system. For example, one change (making some supervisor programs core-resident instead of storing them on disk) seemed to result in a doubling of CPU activity when a software monitor was used to measure performance under actual operation. However, analysis of the accounting data for the same time period indicated that an abnormally heavy load of CPU-bound jobs had caused the shift. Had we not checked the accounting data, we would have made unwarranted conclusions.

Determining Computer Resources Unaccounted for by the Accounting System

Another use of accounting data in conjunction with hardware or software monitors is determining how much of the computer resources are being accounted for and how much are being absorbed by system overhead. An example of this application is illustrated by some tests recently performed at Rand to determine what effect some of the recently implemented on-line systems had on the computer system.† Only CPU utilization was measured, but it was measured both by a software monitor and by analysis of the accounting data. Prior to the full operation of on-line facilities, software-monitor measurements of CPU utilization varied between 10 and 15 percentage points higher than accounting-data measurements. This discrepancy was attributed to general, unrecorded system overhead. However, software-monitor measurements performed during full operation of on-line facilities measured CPU utilization at 85 percent, whereas accounting data during the same time period measured only 32 percent.

†These tests were performed because the resources used by on-line systems at Rand are virtually unaccounted for by the current accounting system.

Therefore, we deduced that during times of high on-line usage as much as 40 percent of CPU time was used by on-line systems.

USE OF ACCOUNTING DATA IN THE DEVELOPMENT AND REVISION OF COMPUTER CHARGING SCHEMES

The use of charging systems on in-house computer systems can provide both a control against overuse of computer resources and an incentive for programmers to write more efficient programs. Another benefit of charging for computer usage is the capability of eliminating possibly marginal data processing that is often run on computers operated as "free goods."

However, in order to reap the benefits of a charging system, the particular scheme used to charge for services must be very carefully formulated. If the computer system is to be operated as a cost-supporting service, it must produce the desired amount of revenue while charging users equitably. Added to the complexity of formulating an effective charging scheme is the economic consideration of charging more for scarce computer resources and less for abundant resources. Without an effective charging scheme, programs that seem efficient to the programmers may run inefficiently on the computer system. An example of an inefficient charging system is one that charges a great deal for memory on a computer system that has a major I/O bottleneck. The result of such a charging system is that users will write programs that require as little memory as possible--but to do this they may have to use overlays to reduce the size of their programs and store all data on tapes and disks instead of in memory. Thus, the memory requested is held at a minimum, which allows more programs into memory (necessary to achieve multiprogramming). However, the I/O operations required to execute the overlays and to transfer the data in and out of core *compound the I/O bottleneck.*

In developing an effective charging algorithm, many installations compromise so that, on the one hand, computer resources are costed-out and, on the other hand, users are encouraged to utilize the system efficiently. One does not necessarily want users to rewrite their programs to maximize use of more abundant resources and minimize use of the resources that are system bottlenecks because such bottlenecks may change over time,

either through system improvements or changes in workloads processed. However, users should be encouraged to write efficient programs and to use discretion when submitting jobs so that they do not overload the system (particularly during the prime shift), for example, by tying up large amounts of core for long periods or running long CPU- or I/O-bound jobs--particularly when one of these resources is a system bottleneck.

Accounting data can be very useful both in the initial formulation of a charging algorithm and in later revisions of the algorithm to maintain its effectiveness in the face of changing user demands, system modifications, and alterations in management objectives for system operation. The accounting data are used in two different ways for this application: (1) as an analysis tool to gain information about the system in the development of alternative charging algorithms, and (2) as input data to test the effectiveness of the alternative charging algorithms.

Use of Accounting Data in the Development of Alternative Charging Schemes

As discussed earlier, accounting data can be very useful (along with other measurement tools) in measuring such performance variables as CPU utilization, I/Os processed per second, and the average number of jobs in memory over a given time period. These measurements can then be used to give an indication of which resources are being underused and which are operating at capacity. Further analysis of the accounting data can determine which workload characteristics have the greatest effect upon the various performance variables. This analysis, combined with the measurements indicating which resources were scarce and which were abundant, can provide an idea of how much weight to assign each workload characteristic in the charging scheme.

An example of how accounting-data analysis can be applied to the formulation or revision of an accounting scheme is illustrated by some analyses done to test the effectiveness of the charging scheme for Rand's IBM 360/65. In Table 5 (a subset of Table 2), "average I/Os used per job step" has virtually no effect upon CPU utilization

(correlation coefficient (r)= -0.17/ -.02 before/after the modification), but has considerable effect upon both I/Os processed per second (r = .65/.66) and job steps processed per second (r = -.60/-.76). Also note the effects of "average CPU seconds used per job step" and "average core requested per job step" on the three performance variables. Examine the correlation coefficients of all three workload characteristics with "average revenue produced per hour" and note the low value for "average I/Os used per job step" in contrast with the relatively high values for the other two. This analysis indicates that the charging algorithm used by Rand may not be charging enough for I/O usage--particularly since the system is considered to be I/O bound.

Table 5

CORRELATION ANALYSIS BETWEEN WORKLOAD CHARACTERISTICS AND PERFORMANCE MEASURES BEFORE/AFTER THE MODIFICATION

Workload Characteristics	Performance Measures			
	CPU Utilization	I/Os Processed/ Second	Job Steps Processed/Hr	Revenue Produced/Hr
Average I/Os used per job step	-.17/-.02	.65/.66	-.60/-.76	.18/.24
Average CPU seconds used per job step	.84/.77	-.17/.16	-.44/-.58	.55/.54
Average core requested per job step	.26/.38	-.33/-.03	-.38/-.48	.51/.45

Use of Accounting Data to Test Alternative Charging Schemes

Although accounting data are useful in developing alternative charging schemes, their most valuable use is as a source of data to test the effectiveness of alternative schemes. The data are easily accessible, require little conditioning or reduction, and are very representative of the data that will be recorded from computer operations in the foreseeable future. The data can be used to evaluate

alternative charging schemes to see (1) if they produce a desired amount of revenue, and (2) how different types of users are charged under the various schemes. Schemes can also be developed to penalize users who abuse the system and to reward users who write efficient programs.

This approach takes the guesswork and risks out of adopting or modifying a charging scheme because it can be fully tested on valid data before implementation.

PROVIDING BACKGROUND INFORMATION FOR A PERFORMANCE-IMPROVEMENT EFFORT

The first phase of a performance-improvement effort (for a computer system) often involves initial data gathering in order to gain an understanding of the system. Such data include workload characteristics and data on the use of various system components, as well as such descriptive information as organizational structures, hardware configuration(s), and software programs in use. For most installations, use of accounting data is not only the easiest way to gain such information but, in many cases, the only way.

For example, accounting-data analysis can provide the following information on workload characteristics:

- Histograms of CPU usage per job.
- Histograms of the number of tapes used per job.
- Charts indicating number of jobs run, CPU seconds used, and number of I/Os used for each language used.
- Percentage of total CPU time used by production jobs.

Resource-utilization information obtainable through accounting data include:

- Core (memory) maps.
- Histograms of CPU utilization over various lengths of time (e.g., over a two-day span, over a month).
- Histograms of the average number of jobs in core.
- Histograms of I/Os executed per second by the system.

IV. CONCLUSIONS

Accounting-data analysis can provide insights leading to major savings or revenue increases on computer systems--at low marginal cost. The accounting data are available at virtually all computer installations; USAF and NASA computer installations should take full advantage of this.

The volume of accounting data available and the fact that they are generated from "typical" workloads make it possible for carefully structured multiple regression and cluster analyses to produce definitive conclusions on computer system performance. Measuring the effect of core augmentation (see Sec. III) is an example. Moreover, the information produced by such analyses (changes in throughput and efficiency) can be used by installation management to evaluate major system modifications with respect to the operating objectives of the installation.

Measurements of system performance or system component utilization taken by hardware or software monitors or generated by simulation or analytical models can be validated through accounting-data analysis. Such analysis can also supply additional information not collected (or, in the case of models, not generated) by these more expensive and sophisticated tools.

Accounting-data analysis can point out major mismatches between incentive structures of current charging algorithms and user contributions to system inefficiency. The correlation analysis on p. 51 is an example. Accounting-data analysis can be very useful in developing and testing a new charging scheme or in updating the current charging scheme in the face of changing workloads or changes in the system.

As future operating systems become more complex, they should expand their accounting systems to collect a wider range of data. IBMs SMF accounting package is an example of such an expanded accounting system. Future operating systems should also have facilities for performing such analyses and for adjusting algorithms to match the needs of individual installations. The Air Force and NASA should include such considerations in future procurement specifications.

Appendix

GRAPHS OF COMPUTER SYSTEM USAGE AND THROUGHPUT CREATED BY ACCOUNTING-DATA PROCESSING PROGRAMS

Figures 15 through 18 present graphs of system usage and throughput created when processing the accounting-log data at Rand. Data points on the graphs correspond to either average values or accumulated values computed every 20 job steps. In other words, a data point corresponds to either an accumulated value for the past 20 job steps processed (e.g., accumulated job steps processed), or an average value during the time period that the past 20 job steps were processed (e.g., high-speed core (memory) in use).

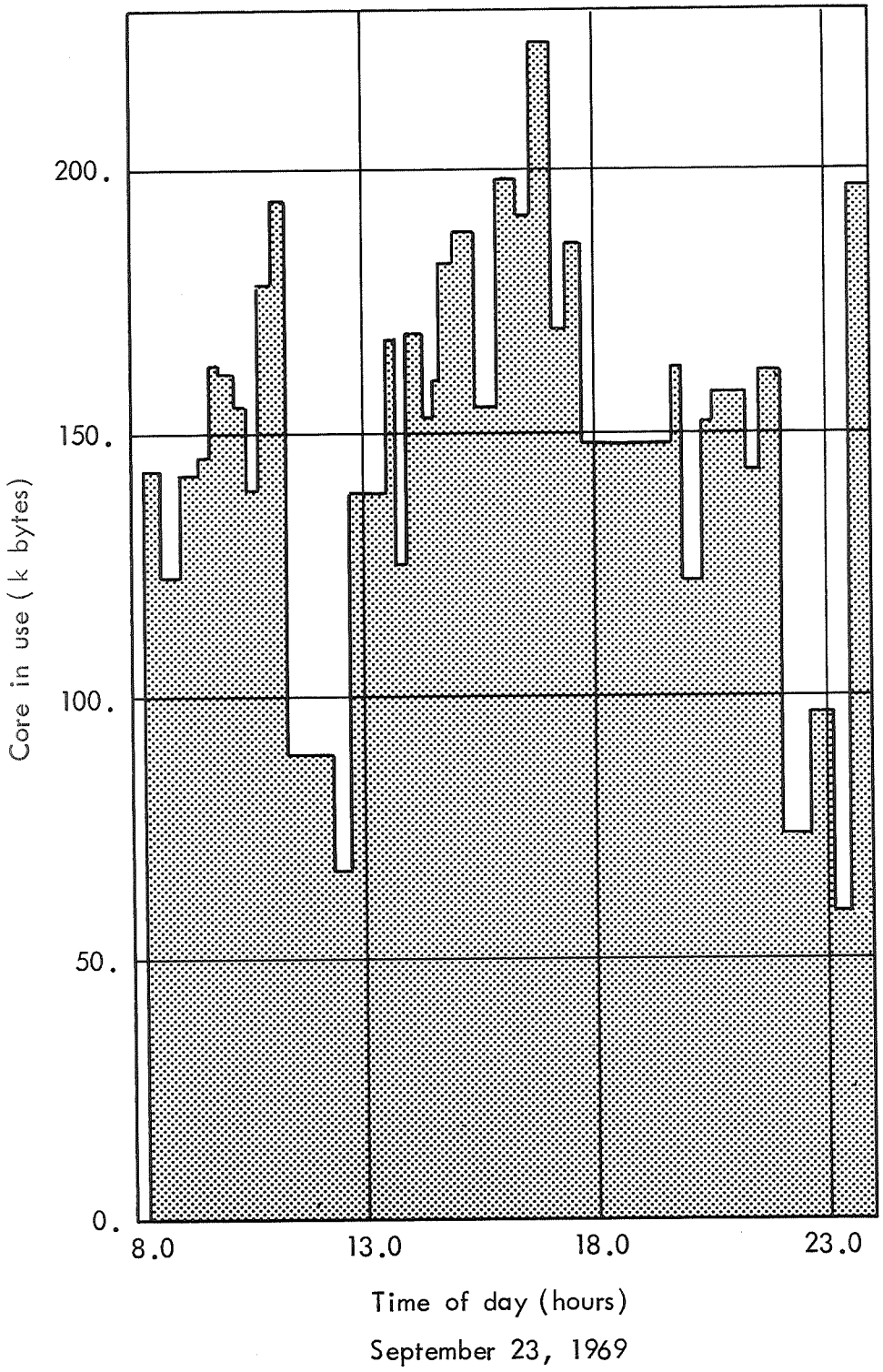

Fig. 15--Core in Use Throughout the Day

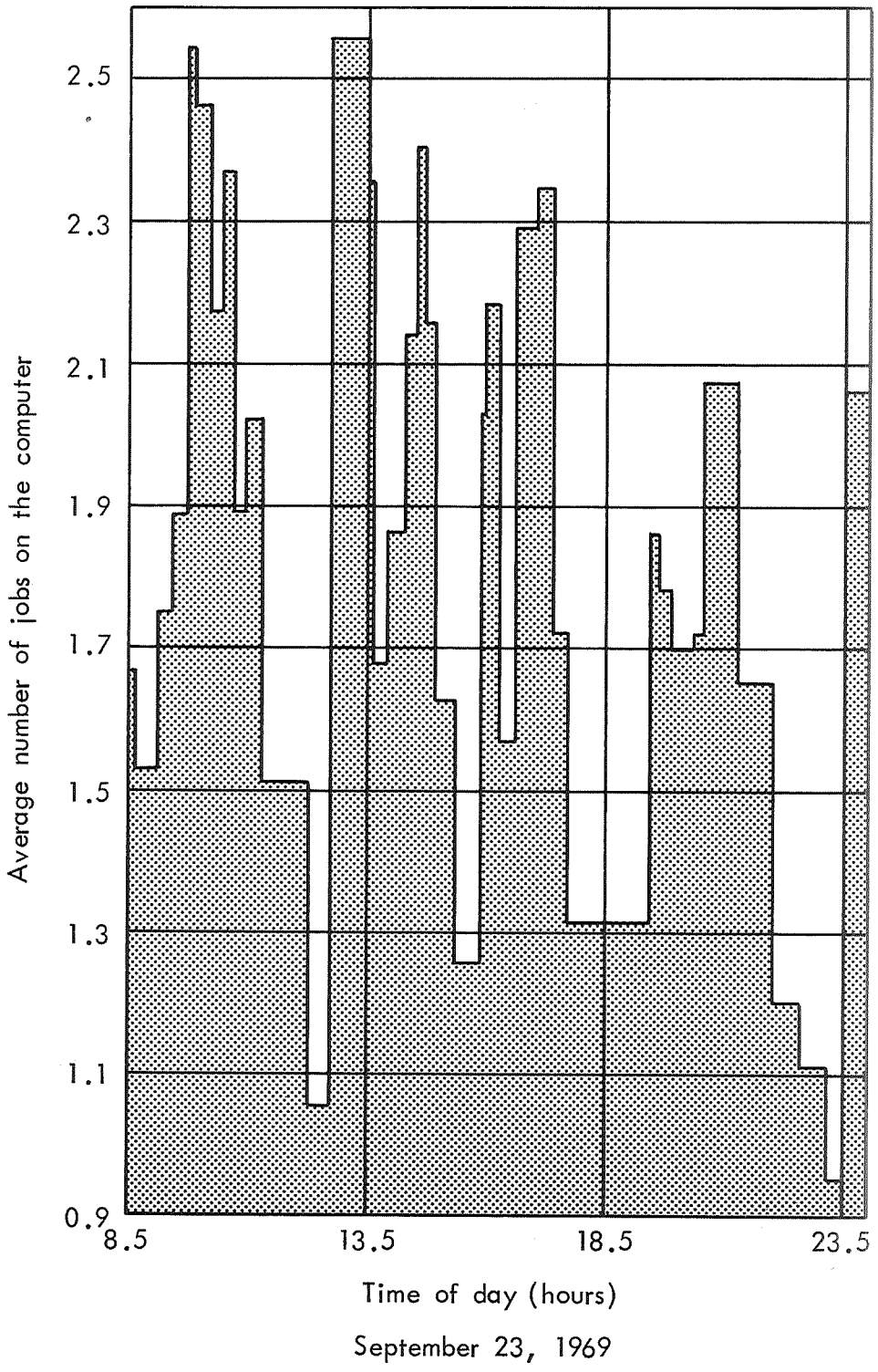

Fig. 16--Average Number of Jobs on the Computer Throughout the Day

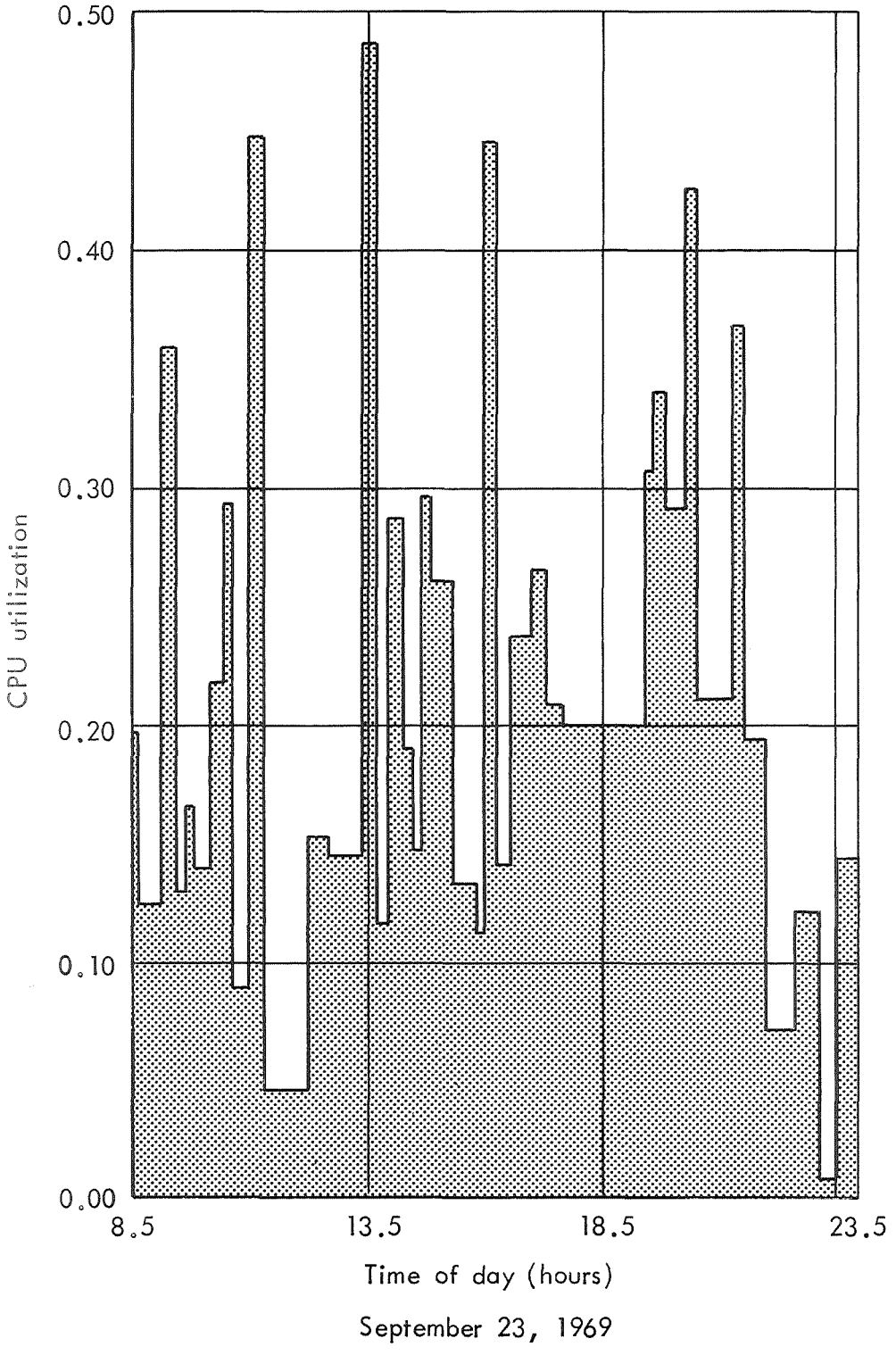

Fig. 17--CPU Utilization Throughout the Day

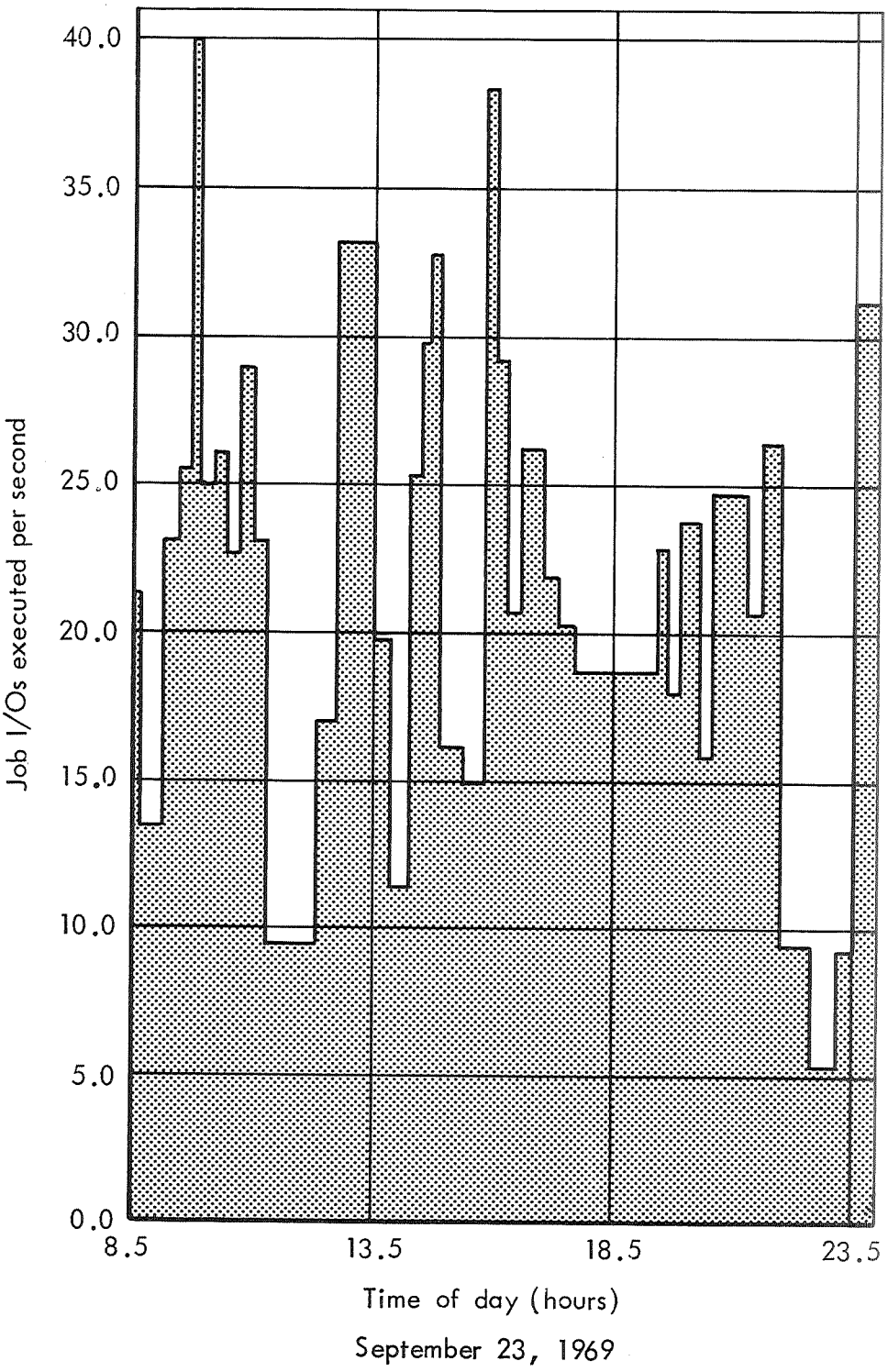

Fig. 18--Job I/Os Executed per Second Throughout the Day

REFERENCES

1. Bell, T. E., *Computer Performance Analysis: Measurement Objectives and Tools*, The Rand Corporation, R-584-NASA/PR, February 1971.

2. *IBM System/360 Operating System: Planning for System Management Facilities*, International Business Machines Corp., Data Processing Division, Form C28-6712-1, White Plains, New York [System Reference Manual].